

***COMPUTERS,
DYNAMICAL SYSTEMS,
PHENOMENA,
AND THE MIND***

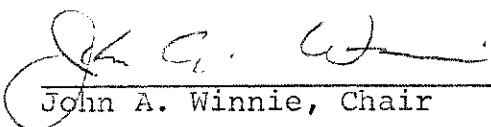
Marco Giunti

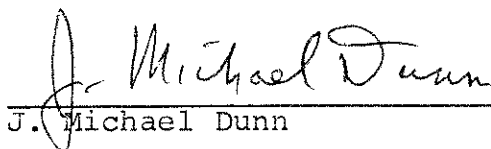
Submitted to the faculty of the Graduate School
in partial fulfillment of the requirements
of the degree
Doctor of Philosophy
in the Department of History and Philosophy of Science
Indiana University

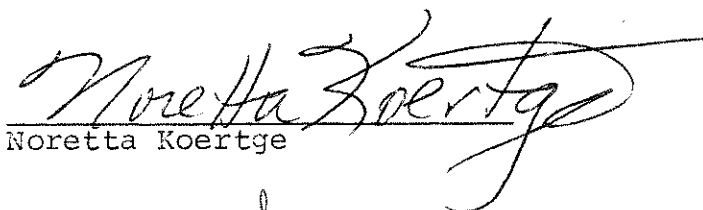
July, 1992

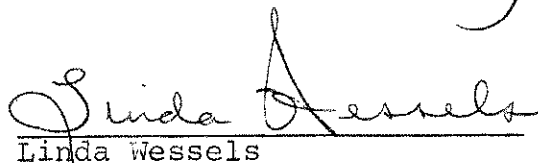
Accepted by the faculty of the Graduate School
in partial fulfillment of the requirements for
the degree Doctor of Philosophy in the Depart-
ment of History and Philosophy of Science,
Indiana University.

Research Committee:


John A. Winnie, Chair


J. Michael Dunn


Noretta Koertge


Linda Wessels

© 1992

Marco Giunti

ALL RIGHTS RESERVED

ACKNOWLEDGEMENTS

My greatest debt is to John Winnie, who provided precious suggestions, sharp criticisms, and continuous encouragement during the whole advancement of my dissertation project. Being his student has been an extraordinary intellectual and personal adventure. I also wish to thank Noretta Koertge, who supervised my earlier studies in the philosophy of science. What I learned from her about scientific problems, explanation, and discovery has been the basis for many of the ideas of chapter 3. Michael Dunn first introduced me to the mysteries of computability theory. In particular, I wish to thank him for the advice concerning chapter 2. Linda Wessels' criticisms prompted me to improve and rewrite several parts of chapter 3. I am especially grateful to her for the advice concerning the semantic conception of theories. Several discussions and exchanges with Tim van Gelder have been extremely important for the final formulation of the ideas of chapter 4. Finally, I wish to thank the participants of the Blunderbuss group: Kevin Korb, Osvaldo Pessoa, Raymundo Morado, Yu-houng, David Chalmers, Gordon Beavers, Gary Curtis, and John Ritner.

ABSTRACT

This work addresses a broad range of questions which belong to four fields: *computation theory, general philosophy of science, philosophy of cognitive science, and philosophy of mind*. *Dynamical system theory* provides the framework for a unified treatment of these questions.

The main goal of this dissertation is to propose a new view of the aims and methods of cognitive science - *the dynamical approach* (chapter 4). According to this view, the object of cognitive science is a *particular set of dynamical systems*, which I call "*cognitive systems*". The goal of a cognitive study is to *specify a dynamical model of a cognitive system, and then use this model to produce a detailed account of the specific cognitive abilities of that system*. The dynamical approach does not limit *a-priori* the form of the dynamical models which cognitive science may consider. In particular, this approach is compatible with both computational and connectionist modeling, for both computational systems and connectionist networks are special types of *dynamical systems*.

To substantiate these methodological claims about cognitive science, I deal first with two questions in two different fields: (1) *What is a computational system?* (2) *What is a dynamical explanation of a deterministic process?*

Intuitively, a *computational system* is a *deterministic system which evolves in discrete time steps, and which can be described in an effective way*. In chapter 1, I give a formal definition of this concept which employs the notions of *isomorphism* between dynamical systems, and of *Turing computable function*. In chapter 2, I propose a more comprehensive analysis which is based on a natural generalization of the concept of Turing machine.

The goal of chapter 3 is to develop a theory of the *dynamical explanation* of a deterministic process. By a "dynamical explanation" I mean the specification of a *dynamical model* of the system or process which we want to explain. I start from the analysis of a *specific type of explanandum -- dynamical phenomena --* and I then use this analysis to shed light on the general form of a dynamical explanation. Finally, I analyze the structure of those theories which generate explanations of this form, namely *dynamical theories*.

TABLE OF CONTENTS

<i>Introduction</i>	1
Chapter 1: DETERMINISTIC DYNAMICAL SYSTEMS AND COMPUTERS	
1. Introduction	15
1.1 Classic computation theory	15
1.2 The symbolic approach in cognitive science	20
1.3 Plan	23
2. Deterministic dynamical systems	26
3. Isomorphic systems and computational systems	32
4. Emulation of a system	41
5. Reversible versus irreversible systems	48
6. Realization of a system	57
7. Virtual systems, the realizability of irreversible systems, and the existence of universal reversible systems	69
8. Appendix	75
Chapter 2: GENERALIZED COMPUTATIONAL SYSTEMS	
1. Introduction	82
2. Pattern fields and generalized Turing machines	93
3. The relation between generalized Turing machines and ordinary ones	105
4. Is a non-recursive pattern field necessary for computing non-recursive functions?	119
5. Generalized computational systems on regular pattern fields	128
Chapter 3: DYNAMICAL PHENOMENA, EXPLANATIONS, AND THEORIES	
1. Introduction	134
2. Dynamical phenomena	147
2.1 Magnitudes and dynamical models	151
2.2 The correspondence between models and systems, and the concept of a dynamical phenomenon	158
2.3 Dynamical studies as attempts to solve dynamical problems ..	161
3. Frameworks and explanations	170
3.1 Frameworks	170
3.2 Dynamical explanations	175
3.3 Explanations as solutions of dynamical problems, and the inductive method	179
4. Principles, laws, and theories	183
4.1 Theoretical principles, specific principles, and laws	183
4.2 Theoretical frameworks and theoretical explanations	188

4.3 Dynamical theories	190
4.4 The deductive method for solving dynamical problems, and the heuristic value of dynamical theories	198
5. Two examples of dynamical theories	205
5.1 The theoretical principles of Impetus theory (or Aristotelian & Impetus dynamics)	206
5.2 The standard models of Impetus theory	211
5.3 The theoretical principles of Newtonian dynamics	212
5.4 The standard models of Newtonian dynamics	214
5.5 Two inconsistent theories which explain the same phenomena	214
5.6 The heuristic values of Impetus theory and Newtonian dynamics	223
Chapter 4: COGNITIVE SYSTEMS AND THE MIND	
1. Introduction	229
1.1 The information processing theory of the mind	231
1.2 Plan	234
2. The problem of meaning	239
2.1 The intension of mental states	241
2.2 The extension of mental states	246
2.3 The intentionality of mental states	250
3. The relation between the body and the mind	254
4. The problem of consciousness	265
5. Is the mind a computational system?	271
5.1 Lucas' argument	271
5.2 Is the information processing theory of the mind consistent with the connectionist approach?	277
6. Cognitive science as a branch of dynamics	284
6.1 Cognitive systems	285
6.2 Cognitive studies as dynamical studies	290
6.3 The symbolic, the connectionist, and the dynamical approach	293
7. Towards a dynamical theory of minds	296
Conclusions	305
Bibliography	308

Introduction

This work addresses a broad range of questions which properly belong to four different fields: *computation theory*, *general philosophy of science*, *philosophy of cognitive science*, and *philosophy of mind*. The first two chapters deal with foundational issues in computation theory. The third chapter discusses the problem of the dynamical explanation of a deterministic process. Finally, the fourth chapter proposes a new view of the aims and methods of cognitive science, and it also discusses some classic issues in the philosophy of mind. I will attempt here to highlight the main ideas of each chapter, and to explain how the specific investigations are internally related to form an organized body.

From the methodological point of view, a single approach underlies the treatment of all specific problems. This methodological approach can be synthetically described as the attempt to formulate and solve each specific question within the framework of dynamical system theory. The basic concept which is at work throughout the whole course of my investigations is that of a *deterministic dynamical system*. Given the importance of this concept, it is useful to briefly introduce it here in an informal manner. This concept expresses in a precise way the basic properties of any *deterministic process*. From the mathematical point of view, a dynamical system is a quite simple structure

composed of three elements. The first element is a set T which represents time¹. The second element is a non-empty set M which represents all the possible states through which the system can evolve². Finally, the third element is a set of functions $\{g^t\}$ which tells us the state of the system at any instant $t \in T$, provided that we know the initial state³. For example, if the initial state is $x \in M$, the state of the system at time t is given by $g^t(x)$, the state at time $w > t$ is given by $g^w(x)$, etc. The functions in the set $\{g^t\}$ must only satisfy two conditions. First, the function g^0 must take each state to itself, for the state at time 0 when the initial state is x obviously is x itself. Second, the composition of any two functions g^t and g^w must be equal to the function g^{t+w} , for the evolution up to time $t+w$ can always be thought as two successive evolutions, the first up to time t , and the second up to time w .

The heuristic principle which I have constantly used is *try to find those parts of a problem which can be represented as a dynamical system*. This general rule turns out to be extremely useful and powerful. The broad applicability of this rule is apparent once we reflect how widespread dynamical processes are. What is surprising to realize, however, is how much we can gain by the simple act of *explicitly thinking of a system, object, or part of the world which changes through time, as a dynamical system*. On the one hand, if we were able to explicitly

¹ T may be either the *reals*, the *rationals*, the *integers*, or the *non-negative portions* of these structures.

² M is called the *phase space* (or sometimes the *state space*) of the system.

³ Each function in $\{g^t\}$ is called a *t-advance* (or a *state transition*) of the system.

describe the possible states of the system, and the possible transitions between these states, we would know exactly how the system evolves. This, however, is an ideal case which rarely occurs. It is more common to have an *implicit* description of the set of possible transitions, for example in the form of a differential equation, and some knowledge about the properties of the set of possible states. *The worst case is when we do not know anything else except that a certain process is a dynamical system.* Still, even in this case, there is a *corpus* of theory and concepts which can be applied to the system. The simple act of representing a dynamical process as a dynamical system allows us to use these abstract tools and, as a rule, this produces a net increase in our understanding.

With regard to the content, I have tried to make each chapter as self contained as possible. However, as a rule, each successive chapter presupposes certain parts of the others. The relation is usually as follows. Each chapter has a specific goal and develops concepts and theory to achieve its goal. Some of the concepts and results which are discussed in detail in the previous chapters are then used in the successive ones. When this is done, I refer to the relevant parts for the details, but I try to give an intuitive explanation of the concepts involved. This should allow an independent reading of each chapter, but such a reading will not permit to fully appreciate the import of certain points.

Even though each chapter can be read as a separate work on a specific subject, the first three chapters provide the conceptual framework for the discussion of the methodological foundations of cognitive science which I develop

in chapter 4. My *methodological proposal* consists in thinking of *cognitive science as a special branch of dynamics*. This means that (i) I identify a *cognitive system* with a dynamical system which is realized by a physical system, and has cognitive abilities (ch. 4, def. 1); (ii) I take the goal of each specific cognitive study to be the production of a *dynamical explanation* (ch. 3, def. 7) of a particular cognitive system. The *dynamical approach* which I propose is thus characterized by three basic tenets: (a) the explicit recognition of the dynamical character of cognition; (b) the disposition to apply methods and concepts from dynamical system theory to the study of *the mental*⁴; (c) the disposition to explore the *whole space* of the dynamical systems in order to locate and map the region of the *cognitive systems*.

The *symbolic approach* to cognitive science maintains that cognitive systems are included in a *special class of computational systems*. This special class can be intuitively characterized as the class of all those computational systems which, somehow, *operate like* Turing machines. The *connectionist approach*, on the other hand, identifies cognitive systems with a *certain class of neural networks*. This class intersects, but is not included in, the class of the computational systems. One of the main theses of this dissertation is that both computational systems and connectionist networks are special types of *dynamical systems*. It thus follows that the *dynamical approach* which I propose is the methodological framework which best allows for a plurality of empirical studies within a unified theoretical perspective. To substantiate these methodological claims about cognitive science,

⁴ I use the term "the mental" to indicate the class of all cognitive systems.

however, I must deal first with two basic questions in two different fields: (1) *What, exactly, is a computational system?* (2) *What, exactly, is a dynamical explanation of a deterministic process?* Both chapter 1 and chapter 2 discuss the first question, while chapter 3 is devoted to the second.

The specific goal of the first chapter is to give a formal analysis of three basic concepts of computation theory: the general notion of a *computational system*, the relation of *emulation* between two computational systems, and the relation of *realization* between a computational system and the concrete system (the hardware) which implements it. There are two possible ways of looking at computation theory. On the one hand, *one can focus on the numbers, functions, or sets* which can be computed or recognized by means of an effective process. The discipline that studies this type of problem is usually called "computability theory", and this name stresses the fact that the primary goal of the theory is to determine which *entities* can be computed. On the other hand, *one can focus on the effective processes themselves*, and ask several interesting questions. The first natural question is: *what is an effective process, and what distinguishes it from other processes that are not effective?* The curious fact is that computer scientists have not given this question a precise answer. Still, this question is very similar to the one which has been investigated with great care: what is an effective algorithm or function, and what distinguishes it from other functions which are not effective?

In chapter 1, I attempt to fill this gap by proposing a theory of what an effective

process is. The strategy that I use consists in thinking of an arbitrary effective process as a special type of dynamical system, which I call a "*computational system*". Intuitively, a computational system is a *deterministic system which evolves in discrete time steps, and which can always be described in an effective way*. This means that there is an effective procedure for determining the possible states of an isomorphic system, and that the state-transitions of this system are effective transformations of finite symbol structures. I give a formal definition of this concept which is based on the notions of *isomorphism* between two dynamical systems, and of *Turing computable function* (ch. 1, def. 3). One of the results which I take to be more interesting is the proof that Turing machines are a special type of computational system (ch. 1, ex. 3.1). This result in fact shows that the process by which a Turing machine computes a function is Turing computable, and it thus partially justifies the concept of Turing computability itself⁵.

Other results of chapter 1 concern the relations of *emulation* and *realization*, and the concept of a *universal system*. Both relations express the intuitive idea of a system which is capable of exactly reproducing the behavior of a second system⁶. Therefore, given a class of systems X, we can define two concepts of

⁵ More precisely, this proof and the 'non-trivial part' of Turing's thesis ([a] all effective functions are Turing computable) provide a partial confirmation of the 'trivial part' of Turing's thesis ([b] all Turing computable functions are effective). In fact, if the process by which a Turing machine computes a function were not Turing computable, we should obviously conclude (by [a]) that the computed function itself is not effective.

⁶ The *emulation* relation is a special case of the *realization* relation. In turn, the relation of *isomorphism* between two dynamical systems is a special case of emulation. The relations of isomorphism, emulation, and realization are defined for *two arbitrary dynamical systems*. The main difference between emulation and realization is that the first relation preserves *the possible types of orbits* of the emulated system, while the second does not. For example, if a system has two

universality: a system is E-universal with respect to X just in case it emulates all systems in class X; a system is R-universal with respect to X just in case it realizes all systems in class X. The interesting fact about these concepts of universality is that *they are not limited to computational systems, but they instead apply to arbitrary dynamical systems*⁷. I will employ this fact in chapter 4, where I will propose a new way of explaining the seemingly unrestricted cognitive abilities of the mind. If the mind is a *dynamical system*, then its *cognitive universality*⁸ can be thought as a special case of either E-universality or R-universality. More precisely, a dynamical system S is cognitively universal if there is a sufficiently broad class of cognitive systems C such that S is E-universal with respect to C or S is R-universal with respect to C.

The definition of a computational system which I give in chapter 1 presupposes that *the class of all effective transformations of finite symbol structures be reducible to the class of the functions computable by a Turing machine*⁹. Chapter 2 discusses in detail this hypothesis, so that the investigations of this chapter concern the foundations of the theory proposed in chapter 1. The main

orbits which merge, and this system is emulated by a second system, this second system will also have two merging orbits. This fact prevents reversible systems from emulating irreversible systems with merging orbits, for no reversible system has merging orbits. Some reversible systems, however, can exactly reproduce the behavior of irreversible systems with merging orbits. The relation which holds in this case is that of *realization*.

⁷ The reason is that the relations of emulation and realization are defined for *two arbitrary dynamical systems* (ch. 1, def. 4, def. 5, and def. 6).

⁸ By "cognitive universality" I mean the property of having many and differentiated cognitive abilities, which allow a system to appropriately perform in a vast range of tasks or situations.

⁹ This hypothesis is traditionally called Turing's thesis.

conclusion which I reach in chapter 2 is that *the concept of an effective transformation of finite symbol structures is not absolute, but instead depends on the relational structure of the infinite support¹⁰ on which the elementary symbols are written and manipulated.* Ordinary Turing machines operate on finite symbol structures (strings) written on an infinite support (a linear tape) whose topography is extremely simple. But one can think of effectively transforming finite symbol structures written on an infinite support whose topography is arbitrarily complicated. It is in fact possible to define *generalized Turing machines* which operate on these 'supertapes' in exactly the same way as ordinary Turing machines operate on linear tapes. The interesting result is that some of these generalized Turing machines are able to compute non-recursive functions. At the end of chapter 2, I will propose a more comprehensive definition of a computational system (a *generalized computational system on a regular pattern field*, def. 8), which is based on the concept of a generalized Turing machine. The definition of a *computational system* which I give in chapter 1 (def. 3) turns out to be a special case of the new one. In fact, generalized computational systems on a regular pattern field reduce to computational systems when the pattern field is identified with the tape of an ordinary Turing machine. Finally, I also prove that any generalized Turing machine which operates on a regular pattern field is a generalized computational system on that pattern field (ch. 2, th. 6). This theorem thus extends the analogous result for ordinary Turing machines (ch. 1, ex. 3.1).

¹⁰ I call this support a "pattern field" (ch. 2, def. 1).

The goal of chapter 3 is to develop a general theory of the *dynamical explanation* (ch. 3, def. 7) of a deterministic process. By a "dynamical explanation" I mean the specification of a *dynamical model* (ch. 3, def. 2 and def. 4) of the system or process which we want to explain. Historically, the first dynamical explanations were provided by Galileo, who studied in detail the free fall of a body, the motion of a sphere on an inclined plane, and the motion of a projectile. Other classic examples of dynamical explanations are those of the motion of a pendulum, and of the revolution of the planets about the sun. Dynamical explanations were first put forth by physicists but, since then, many other sciences have been concerned with this type of explanation. *The main thesis of this dissertation is that cognitive science is one of these, for I take the goal of each cognitive study to be the specification of a dynamical model of a particular cognitive system.*

Even though the literature on scientific explanation is huge, *dynamical explanations* have not been studied in detail so far. Scientific explanation is in fact traditionally studied from a very general point of view, which does not pay much attention to the specific structure of the *phenomena*¹¹ which scientists attempt to explain. Chapter 3 reverses this approach. I start from the analysis of a *specific type of explanandum -- dynamical phenomena*¹² -- and I then use this analysis to shed light on the general form of a *dynamical explanation*. Finally, I analyze the

¹¹ I use the term "phenomenon" in a sense which is consistent with the Kantian tradition: any system or process which can be subsumed under certain categories.

¹² By a "dynamical phenomenon" I mean any deterministic process or system which can be described by means of a finite number of interdependent magnitudes (ch. 3, def. 5).

structure of those *theories* which generate explanations of this form, namely *dynamical theories* (ch. 3, def. 10). The only concept from the previous chapters which I use in chapter 3 is that of a deterministic dynamical system. On the other hand, the concepts developed in chapter 3 (together with those of chapter 1) provide the necessary framework for the discussion of the aims and methods of cognitive science, which is the main topic of chapter 4.

In chapter 4, I focus first on *the information processing theory of the mind*, and I then propose an alternative methodological view. The information processing theory of the mind maintains that *the mind is a computational system realized by a concrete physical system*. This basic assumption, together with the further hypothesis that the mind is a *universal computer*¹³, is the foundation of a definite research program. Since by assumption the mind is a universal computer, its operations depend on the programs which are stored in its working memory. Different programs specify different computational systems, and these systems have particular cognitive abilities appropriate for specific situations or tasks. Therefore, *the aim of a concrete empirical research is to specify a dynamical model of one of these special purpose computational systems*. This can be obtained by writing a computer program which specifies a second computational system, and this computational system must *emulate* the special purpose

¹³ Universal computers have a working memory in which both *programs* and *data* can be stored, and they have the capacity to execute *any* program stored in their memory. Since any computational system can be specified by a program, universal computers are able to emulate any computational system. Universal computers are thus E-universal with respect to the class of all computational systems. The classic example of a universal computer is a universal Turing machine.

computational system which we want to explain.

The first question I discuss in chapter 4 is whether this view of the aims and methods of cognitive science is justified. In this connection, I consider a number of arguments which attempt to show that it is *in principle impossible* to identify the mind with a computational system realized by a concrete physical system. These objections are of two types. The first type singles out a certain mental property P, and then attempts to show that the identification of the mind with a computational system realized by a concrete physical system entails that the mind cannot have property P. In particular, I consider whether the information processing theory of the mind can in principle explain the intentional character of mental states¹⁴, the existence and nature of conscious states¹⁵, and the ability of recognizing the truth of the Gödel sentence of an arbitrary formal theory of arithmetic¹⁶. Several authors have variously maintained that the identification of the mind with a computational system realized by a physical system is inconsistent with these three properties of concrete minds. The general thesis I defend is that these arguments fail to make their point, and that it is *in principle possible* to identify the mind with a computational system realized by a concrete physical system.

The second type of objection has been recently raised by Searle (1990b).

¹⁴ Searle has claimed that it cannot (1980, 1984, 1990a).

¹⁵ Maudlin (1989) has recently put forth an ingenious argument to show that any computational theory of consciousness must be inconsistent.

¹⁶ Lucas (1961, 1968a, 1968b) has maintained that, since the mind has this ability, it cannot be a computational system. A similar argument has been recently put forth by Penrose (1989).

Searle claims that the question whether a physical system realizes a computational one is not factual but, rather, a matter of conventions. If Searle were right, the information processing theory of the mind would face a serious problem, for the proponents of this approach maintain that the mind is a computational system realized by the brain, and they take this hypothesis to be a *synthetic* one. I will show in section 6 of chapter 1 that the realization relation between two dynamical systems does not involve any conventional element, for it depends on the *existence* of a mapping between the *states* of the realized system and *sets of states* of the realizing one. The relation between the mind and the brain postulated by the information processing theory is in fact a special case of the realization relation between two dynamical systems. Therefore, Searle's claim about the conventional character of this relation is unjustified.

Even though it is *in principle* possible to identify the mind with a computational system realized by a concrete physical system, we should also consider whether this hypothesis is consistent with the most recent developments in cognitive science. I will argue that it might not be, provided that the mind can be identified with a finite *neural network with continuous activation levels* (ch. 4, sec. 5.2). Furthermore, the question of whether this identification is possible is an *empirical* one. This argument is based on the analysis of a *computational system* which I develop in chapters 1 and 2, and it can in fact be generalized to any *continuous*

*dynamical system*¹⁷.

The possibility that the mind could be identified with a *non-computational* dynamical system thus is a real one. Therefore, the methodological perspective provided by the information processing theory of the mind is too restrictive, for it rules out *a-priori* a broad range of possibilities which instead should be empirically investigated. My proposal for overcoming this methodological impasse is a more comprehensive theory - *the dynamical approach* (ch. 4, sec. 6) - which does not assume that *the mind* is the primary object of cognitive science, and does not presuppose the *computational* character of this object. The basic idea of this methodological view is that the *primary* object of cognitive science is not the mind but, instead, a *particular set of dynamical systems*, which I call "*cognitive systems*". A second name which is also appropriate is "*the mental*". I prefer the first term for it more clearly brings about the shift of perspective between this view and the one which primarily focuses on *the mind*. According to this view, *cognitive systems* are all those *dynamical systems* which are realized by physical systems and have *cognitive abilities*. The goal of a *cognitive study* is to specify a *dynamical model* of a particular cognitive system, and to produce a *detailed account of the specific cognitive abilities of this system*.

An important feature of the dynamical approach is that it does not limit *a-priori* the form of the dynamical models which can be considered. For example, some

¹⁷ By a "*continuous dynamical system*" I mean a dynamical system $\langle T, M, \{g^t\} \rangle$ such that at least one of the following conditions is satisfied: (i) the time T of the system is the set of the real numbers (or the set of the non-negative reals); (ii) the phase space M is not denumerable.

of these models are *computational*, for they are specified by means of computer programs. Others consist of *neural networks*, which are specified by their connections, weights, and by the input-output characteristics of each unit. Finally, a third type of models are those specified by *systems of differential equations*. The dynamical approach to cognitive science can thus be characterized by three basic tenets: **(a)** the explicit recognition of the dynamical character of cognition; **(b)** the disposition to apply methods and concepts from dynamical system theory to the study of *the mental*; **(c)** the disposition to explore the *whole space* of the dynamical systems in order to locate and map the region of the *cognitive systems*.

In the last section of chapter 4, I finally sketch a broad outline of a theory of minds consistent with the dynamical approach. In the first place, *I think of minds as a special type of cognitive systems which satisfy at least four further properties: (1) are realized by concrete physical systems; (2) have intentional states; (3) have conscious states; (4) are cognitively universal*. I then ask whether a *dynamical theory of minds*¹⁸ can *in principle* explain these properties. My thesis is that it can, and that these explanations are at least as good as those provided by the information processing theory of the mind. In fact I take them to be superior, for they do not *presuppose* the *computational* character of the mind, but only the *weaker hypothesis* that minds are *dynamical* systems realized by concrete physical systems.

¹⁸ By a "dynamical theory of minds" I mean any theory which entails the hypothesis: minds are dynamical systems realized by concrete physical systems.

Chapter 1

DETERMINISTIC DYNAMICAL SYSTEMS AND COMPUTERS

1. Introduction
 - 1.1 Classic computation theory
 - 1.2 The symbolic approach in cognitive science
 - 1.3 Plan
 2. Deterministic dynamical systems
 3. Isomorphic systems and computational systems
 4. Emulation of a system
 5. Reversible versus irreversible systems
 6. Realization of a system
 7. Virtual systems, the realizability of irreversible systems, and the existence of universal reversible systems
 8. Appendix
-

1. Introduction

The aim of this chapter is to provide a formal analysis of three concepts which are at the heart of both computation theory and cognitive science: the general notion of a *computational system*, the relation of *emulation* between computational systems, and the relation of *realization* between a computational and a physical system.

1.1 Classic computation theory

Effective procedures are traditionally studied from two different and complementary points of view, the function-approach and the system-approach.

The function-approach is concerned with the question of individuating those numeric functions which are effectively calculable. This approach has reached its systematization with the theory of the partial recursive functions (Gödel, Church, Kleene). This theory is not directly concerned with computing devices or computations. Rather, the effective calculability of a partial recursive function is guaranteed by the algorithmic nature of its definition.

The system-approach focuses on a group of abstract mechanisms, which are then typically used to compute numeric functions. These devices can be divided in two broad categories: automata or machines (Turing and Post), and systems of rules for symbol manipulation (Post). Below are some of the mechanisms which have been studied:

a. AUTOMATA OR MACHINES

- [1] gate-nets and McCulloch-Pitts nets
- [2] finite automata (Mealy and Moore machines)
- [3] push-down automata
- [4] stack automata
- [5] Turing machines
- [6] register machines
- [7] Wang machines
- [8] cellular automata

b. SYSTEMS OF RULES

- [9] monogenic production systems in general
- [10] monogenic Post canonical systems
- [11] monogenic Post normal systems
- [12] tag systems.

All these mechanisms have much in common, and the system-approach has been traditionally interested in studying the relations between each kind of device and the others, and in establishing what class of numeric functions each

mechanism can compute. Accordingly, two kinds of theorem¹ are typically proved:

- (a) it is shown that systems of a given kind *emulate* systems of another kind (examples: Turing machines emulate register machines and cellular automata, cellular automata emulate Turing machines, etc.);
- (b) systems of a certain kind are proved to be complete relative to the class of the partial recursive functions, that is, the systems of this kind compute all and only the partial recursive functions (examples of complete systems: Turing machines, register machines, cellular automata, tag systems, etc.).

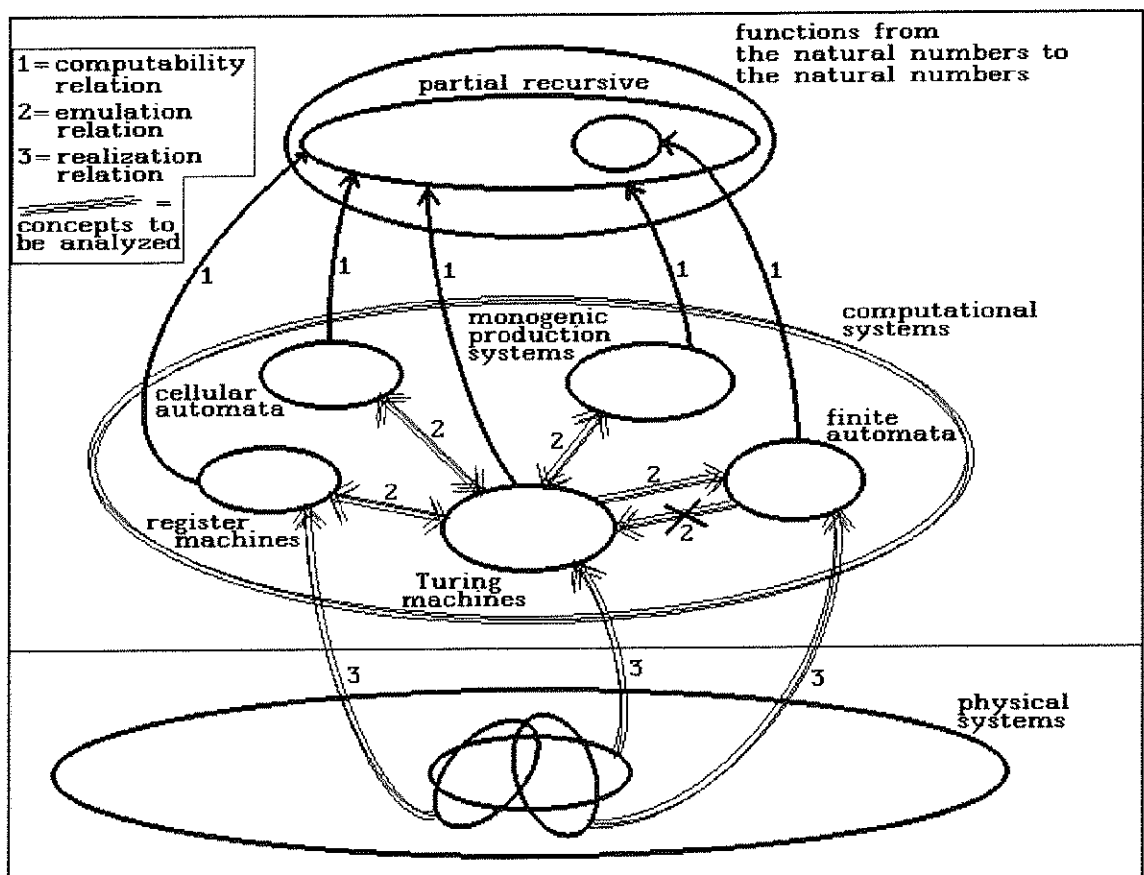


Figure 1 Classic computation theory

¹ Theorems of the first kind are often lemmas to theorems of the second kind. Sometimes, the proof that a certain type of system emulates another type is simply given as part of the proof that the first type can compute all partial recursive functions.

Nevertheless, it is not at all clear what, exactly, all these mechanisms share. The received view is that they are all different types of *computational systems*. It is usually said that *a computational system is a deterministic system which evolves in discrete time steps, and that it can be described in an effective manner*. While this informal characterization is certainly useful, and it conveys a concept which delimits the field of *classic computation theory*², it is clear that this concept is not sufficiently rigorous. It seems that we face here a typical example of a formal problem (Giunti 1988, 431). A precise definition of what a computational system is would give us a deeper, and more unified, insight into the whole field of computation theory and its applications.

I have mentioned above that one of the typical theorems of computation theory consists in showing that, for each device of a first type, there is a device of a second type which is able to *emulate* the first device. If we look at several instances of these proofs, we discover two things. First, in many cases, the emulation relation is not formally defined. Second, it is not clear whether the

² Several authors have studied computational devices whose basic operations are transformations of real numbers or, even more generally, transformations which define some abstract mathematical structure on some set (Blum & al. 1989; Friedman 1971; Shepherdson 1975, 1985, 1988). These devices may not be computational systems in the classic sense, for an operation on real numbers involves the manipulation of an infinite amount of information, and so it may not in general be possible to effectively describe the basic operations of these machines. Within classic computation theory it is possible to define computable functions on the real numbers, but the values of these functions can never be exactly computed. Rather, these values are calculable to any desired degree of accuracy (Turing 1965; Pour-El and Richards 1989). By contrast, the basic operations of the devices mentioned above are assumed to have infinite precision, in the sense that they yield the *exact* values of a real-valued function. Other devices which may not be classic computational systems are *unrestricted* cellular automata. These cellular automata either operate on arrays with an infinite number of cells in a non-blank state, or they do not satisfy the restriction that a neighborhood whose cells are all in the blank state never changes the state of its central cell.

relation which is proved to hold in different cases is in fact the same relation. For example, it can be proved that cellular automata emulate Turing machines and that, conversely, Turing machines emulate cellular automata. The emulation relation between cellular automata and Turing machines, however, has not been adequately defined³, and it turns out that cellular automata emulate Turing machines in a way which is *prima facie* different from the way Turing machines emulate cellular automata⁴.

The *emulation* relation is also usually invoked to introduce the concept of a *virtual system*, which is one of the basic concepts of both computation theory and cognitive science. Whenever a computational system S_1 emulates a second computational system S_2 , it is claimed that it is possible to define a third system S_3 in terms of S_1 , and that S_3 turns out to be essentially the same system as S_2 . The system S_3 is called a virtual system, and this story is usually told in order to explain how different computational architectures can be implemented by the same computer. However, since no general concept of emulation is defined, no general theorem which ensures the existence of a virtual system can be proved.

We encounter a third formal problem when we consider the relation between the abstract mechanisms studied by the system-approach and the physical

³ Alvy Ray Smith defines the emulation relation between cellular automata and Turing machines (1971, def. 4). However, his definition only applies to cellular automata emulating Turing machines, not to the converse case.

⁴ The main difference is that cellular automata are able to emulate each step of a Turing machine in real time, while Turing machines take several steps to emulate one step of a cellular automaton. Furthermore, a Turing machine will in general emulate different steps of a cellular automaton in different times.

systems which supposedly implement or *realize* them. This relation is never explicitly defined, but several facts about it are usually assumed. For example, it is usually believed that computational systems are in principle realizable by physical systems. The informal account of how a physical system would realize a computational one typically goes like this: some of the states of the physical system are identified with the states of the computational one, so that each transition of this system corresponds to a transition between physical states. This is a very plausible idea, but a simple observation shows that it needs some refinement. Many physical systems are reversible, while most computational systems are irreversible. In reversible systems no transition can take two different states to the same state, while this is exactly what happens in irreversible systems. Therefore, transitions between computational states cannot in general be identified with physical transitions. This observation and other anomalies⁵ thus suggest that a formal analysis of the realization relation is badly needed⁶.

1.2 The symbolic approach in cognitive science

The symbolic approach in cognitive science maintains that some *computational*

⁵ A second fact about the realization relation which is assumed without proof is its transitivity with respect to emulation: if S realizes C' and C' emulates C, then S realizes C. This is the principle which is usually invoked to explain the multiple realizability of a computational system C. C can be realized by many physical systems because, in order to concretely realize C, we only need to implement a second computational system, C', which emulates C.

⁶ Unless we are willing to accept that irreversible computational systems are not in principle realizable by reversible physical systems.

systems⁷ have cognitive abilities, and that the brain (or the nervous system) concretely implements, or *realizes*, these systems. This basic principle is thus the hard-core of a definite research program.

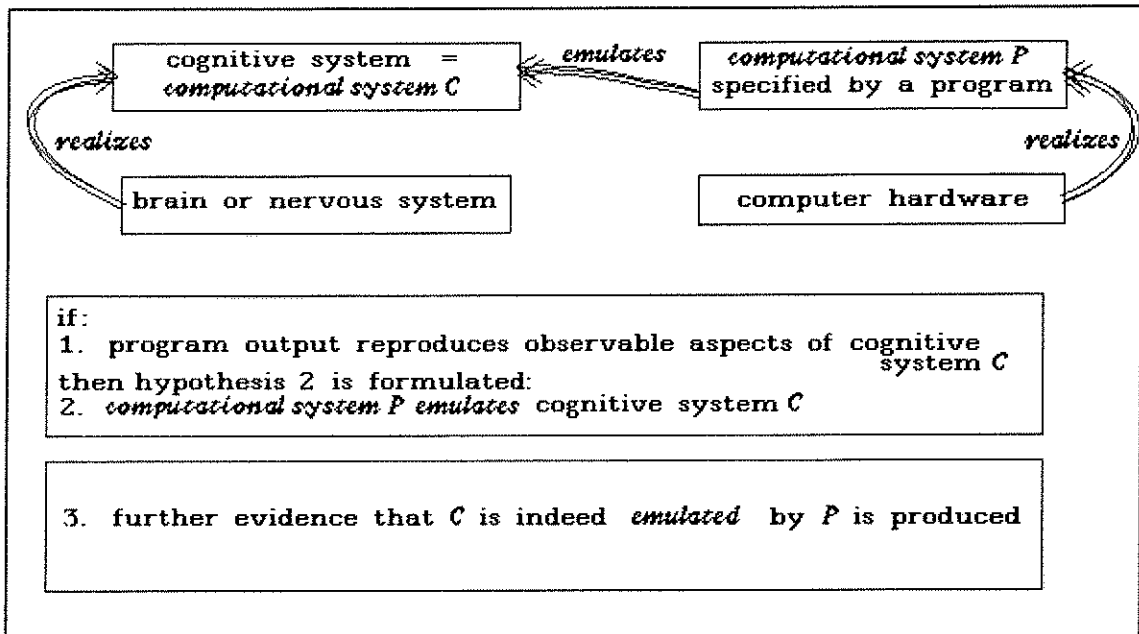


Figure 2 The symbolic approach in cognitive science

Whenever we are interested in studying some cognitive ability (say, for example, language understanding) we should try to determine the computational system C which has this ability and is concretely realized by the brain. This, in turn, will be done in three steps. First, we will try to write a computer program whose performance cannot be distinguished from the performance of a subject with that ability⁸. Second, we will formulate the hypothesis that the computational

⁷ Sometimes, the words "program", "digital computer", "information processing system", etc. are instead used. "Computational system" is the most appropriate term, because it encompasses the intended reference of all other terms.

⁸ That is, the program should pass a Turing-test (Turing 1956) limited to the particular ability that we are studying.

system P specified by the program *emulates* the computational system C which has that ability and is *realized* by the brain. Third, we will seek further evidence that the system P *does indeed emulate* C. This evidence may be of two different kinds: based on psychological experiments, or on knowledge of the neuro-physiological structure of the brain⁹.

It is thus clear that, to understand the methodological foundations of the symbolic approach, we need an adequate analysis of such basic concepts as *computational system, emulation, and realization*. This analysis is in fact necessary for giving a definite meaning to at least two assumptions on which this approach rests. First, that computational systems specified by computer programs *may emulate* computational systems with arbitrary cognitive abilities¹⁰. Second, that these systems are *realized* by the brain. The clarification of these assumptions in turn implies that we formally define the three concepts of *computational system, emulation, and realization*. This is precisely the goal of this chapter. Searle has recently expressed a similar concern:

There is little theoretical agreement among the practitioners on such absolutely fundamental questions as, What exactly is a digital computer? What exactly is a symbol? What exactly is a computational process? Under what physical conditions exactly are two systems implementing the same program? (Searle 1990, 24)

Searle raises these questions to conclude that the problem of whether the

⁹ It should also be noticed that, if C is indeed emulated by P, the relation between the brain and C is exactly the same as the relation between the computer hardware and C. This is implied by the principle: if a system S realizes P and P emulates C, then S realizes C.

¹⁰ As far as we know, computational systems specified by programs and computational systems which have cognitive abilities may be of very different kinds, so that it is not immediately clear in what precise sense systems of one kind may emulate systems of the other kind.

brain realizes a computational system is not a factual one but, rather, a matter of conventions. The results of sections 3, 4, 6, and 7, however, imply that his conclusion is unjustified. I will explicitly discuss Searle's argument in chapter 4.

1.3 Plan

Dynamical system theory provides the natural mathematical framework for carrying out this analytical task. Cellular automata are devices which have already been fruitfully studied from this point of view (Wolfram, 1983a; 1984c; Farmer, Toffoli, and Wolfram 1984; Wolfram, ed. 1986). However, since all computational systems are deterministic systems, they all fall under the scope of dynamical system theory. This discipline is traditionally interested in the study of those dynamical systems which arise from the solution of differential or difference equations. It is, however, possible to define a *dynamical system* in such a way that it captures the idea of an *arbitrary deterministic process*. Once this definition is adopted, it is quite obvious that all known computational devices are dynamical systems, and looking at them under this light makes it possible to analyze the concepts of *computational system*, *emulation*, and *realization* in a completely general and extremely natural manner. The detailed plan is as follows.

Section 2 states the definition of a *dynamical system* and introduces the concept of a *cascade*, that is, a dynamical system with discrete time. This is a crucial concept, for *I will later define a computational system as a cascade which can be described in an effective manner.*

Section 3 carries out the analysis of the concept of a *computational system*. I introduce first the relation of *isomorphism* between dynamical systems. Two systems which satisfy this relation are equivalent from the point of view of their dynamical behavior or, in other words, they can be identified as far as their dynamical properties are concerned. I then define computational systems by means of this relation, of the concept of a cascade, and of the concept of a *Turing computable function*. This definition presupposes the reducibility of the class of all effective transformations of finite symbol structures to the class of the Turing computable functions. This hypothesis is traditionally called Turing's thesis. In chapter 2, I will propose a natural generalization of this thesis, and I will then give a more comprehensive definition of a computational system¹¹.

Section 4 analyzes the *emulation* relation, and proves that this relation is a quasi-ordering (reflexive and transitive) on the set of all deterministic dynamical systems.

Section 5 introduces the distinction between reversible and irreversible systems, and studies the possible types of orbits and t-advances (or state transitions) in both kinds of systems. I define three mutually exclusive and exhaustive classes of irreversible systems: *quasi-reversible*, *weakly irreversible*, and *strongly irreversible*. Strongly irreversible systems are the only systems with merging orbits, and it turns out that most computational systems are in this class.

These results set the stage for Theorem 6 of the next section. This theorem

¹¹ I call a system which satisfies this more general definition a "generalized computational system on a regular pattern field" (see ch. 2, def. 8).

and the definition of the emulation relation entail the important consequence that *all universal computational systems are strongly irreversible, and that they have all possible types of orbits: periodic, eventually periodic, aperiodic, and merging.* Theorem 6 also shows that the *realization* relation cannot be identified with the concept of emulation. I then turn to the formal analysis of this relation. I give two equivalent definitions. The first definition allows me to show that a physical realization of a Turing machine, as usually described, does indeed realize the dynamical system which defines the Turing machine. I then use the second definition to prove that the realization relation is reflexive and transitive.

In section 7, I prove two general results. I first show that, whenever a system S_1 either emulates or realizes a second system S_2 , it is possible to define a *virtual* S_2 by means of the states and the t -advances (or state transitions) of S_1 , and that this system is isomorphic to S_2 . Finally, I prove that, given an arbitrary irreversible system S , it is always possible to construct a corresponding reversible system which realizes S . An immediate and important consequence of this theorem is the *existence of universal reversible systems*¹².

¹² This does not contradict the result of section 6, that all *universal computational* systems are strongly irreversible. Universal systems can be defined in two ways, depending on whether they *emulate* or *realize* all computational systems. If the first definition is chosen, all universal systems turn out to be strongly irreversible. The second definition, instead, allows for universal reversible systems.

2. Deterministic dynamical systems

A property which is shared by all (standard) computational devices is that they are deterministic systems. Intuitively, this means that the future time evolution of the complete (or total) state of the system is determined by the state at the present time. Dynamical system theory allows us to precisely define a deterministic system. Such a structure is an ordered triple $\langle T, M, \{g^t\} \rangle$, where T is a set which represents time, M is the set of all total states of the system, and g is a function from $T \times M$ to M . Intuitively, a total state completely describes the system at some

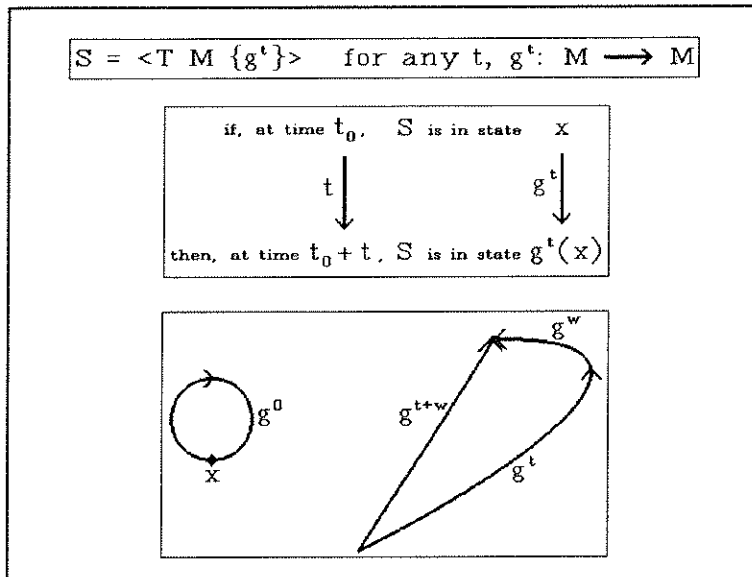


Figure 3 Deterministic dynamical systems

instant, in the sense that it contains all the information sufficient (and necessary) for determining the future (and past) behavior. The function $g(t, x)$ precisely expresses this idea. In fact, once an initial total state x is fixed, $g(t, x)$

tells us the total state which the system will reach at an arbitrary time t . If we take t to be a parameter, the function $g(t, x)$ can be identified with a set of functions $\{g^t(x)\}$ from M to M . Each of these functions is called a t -advance of the system. The t -advances of a dynamical system should satisfy two obvious properties. First,

the t-advance g^0 should be the identity function, for the state reached after 0 time when the system starts in state x obviously is x itself. Second, the composition of any two t-advances g^t and g^w should be equal to the t-advance obtained by adding t and w , for the evolution of the system up to time $t+w$ can always be thought as two successive evolutions, the first up to time w , and the second up to time t . That is, $g^t(g^w(x)) = g^{t+w}(x)$. These properties are all expressed by the following definition:

Definition 1 (deterministic dynamical system)

$S = \langle T, M, g(t, x) \rangle$ is a deterministic dynamical system iff:

- (1) T is either Z, Z^+, Q, Q^+, R, R^+ , where $Z, Q,$ and R are, respectively, the integers, the rationals, and the reals, and $Z^+, Q^+,$ and R^+ , are the non-negative integers, rationals, and reals;
- (2) M is a non empty set;
- (3) g is a function from $T \times M$ to M such that, for any $t \in T$, the functions $g^t: M \rightarrow M$ defined by $g^t(x) =: g(t, x)$ satisfy:
 - (a) g^0 is the identity function on M ;
 - (b) $g^{t+w} = g^t \circ g^w$ (\circ is the composition operation and g^w is applied first)¹³.

The intuitive meaning of this definition is as follows. T represents time, which can either be discrete (Z or Z^+), dense (Q or Q^+), or continuous (R or R^+), and M is the set of all possible total states of the system S we are studying. A total state

¹³ This definition is, essentially, the one given by Arnold (1977, 4). See also Szlenk (1984). The only difference between my definition and that of Arnold is that I allow time = T to be either $R, R^+, Q, Q^+, Z,$ or Z^+ , while Arnold exclusively considers the case $T = R$. Traditionally, dynamical system theory is concerned with those dynamical systems which arise from the study of differential or difference equations. As a consequence, the phase space of these systems is not a bare set, but it has a richer structure (a differentiable manifold or, at least, a metric space). As Arnold makes clear, this further structure is not relevant if we are interested in a formal characterization of any possible deterministic process.

completely describes the system S at some instant. M is called the phase space of S , and $g(t, x)$ has two possible interpretations. If we take t to be a parameter, we obtain a set of functions of the variable x , $\{g^t\}$. Suppose that, at the present time t_0 the state of S is x (relative to some initial state y); then $g^t(x)$ is the state of the system at time t_0+t (relative to the same initial state y). For this reason each function g^t is called a t-advance, or a state transition, of S . If, instead, we take x to be a parameter, we obtain a set of functions $\{g^x\}$ of the variable t . Each of these functions represents the time evolution of the state of the system when the state at time 0 is x . Each function g^x is thus called a state evolution, or a motion, of S . It thus follows that $g(t, x)$ can either be interpreted as the set of all possible t-advances or as the set of all possible state evolutions of S . For any $x \in M$, the image of g^x is called a phase curve in M , or the orbit of x , which is abbreviated $\text{orb}(x)$. The phase portrait of S is the set of all orbits of S . The extended phase space of S is the set $T \times M$, and each state evolution (or motion) of S is also called an integral curve in extended phase space.

Example 1.1 (the Galilean model of free fall)

A very simple example of a deterministic dynamical system is the Galilean model of free fall $S = \langle T, Y \times V, \{g^t\} \rangle$, where: T is time, Y are the vertical positions of a freely falling body, V the vertical velocities, and $g^t: Y \times V \rightarrow Y \times V$ is defined by: $g^t(y, v) = \langle (y + vt + 1/2ct^2), (v + ct) \rangle$.

Let us verify that conditions (3a) and (3b) of definition 1 hold. We must prove:

(a) $g^0(y, v) = \langle y, v \rangle$

(b) $g^{t+w}(y, v) = g^t(g^w(y, v))$

from the definition of g :

1. $g^0(y, v) = \langle (y + v \cdot 0 + 1/2c \cdot 0^2), (v + c \cdot 0) \rangle = \langle y, v \rangle$ // (a) is proved //

from the definition of g :

1. $g^{t+w}(y, v) = \langle (y + v(t+w) + 1/2c(t+w)^2), (v + c(t+w)) \rangle$

from the definition of g :

$$\begin{aligned}
 2. \quad g^t(g^w(y \ v)) &= g^t((y + vw + 1/2cw^2) (v + cw)) = \\
 &<(y + vw + 1/2cw^2 + (v + cw)t + 1/2ct^2) (v + cw + ct)> = \\
 &<(y + vw + 1/2cw^2 + vt + cwt + 1/2ct^2) (v + c(t+w))> = \\
 &<(y + v(t+w) + 1/2c(w^2 + 2wt + t^2)) (v + c(t+w))> = \\
 &<(y + v(t+w) + 1/2c(t+w)^2) (v + c(t+w))>
 \end{aligned}$$

from 1 and 2:

$$3. \quad g^{t+w}(y \ v) = g^t(g^w(y \ v)) \quad //(\text{b) is proved}//.$$

Definition 1 entails the following properties:

If $T = \mathbb{Z}, \mathbb{Q},$ or \mathbb{R} , then:

- (1) $\{g^t\}$ is a commutative group with respect to the composition operation \circ . The unity is g^0 and, for any $t \in T$, the inverse of g^t relative to \circ is g^{-t} ;

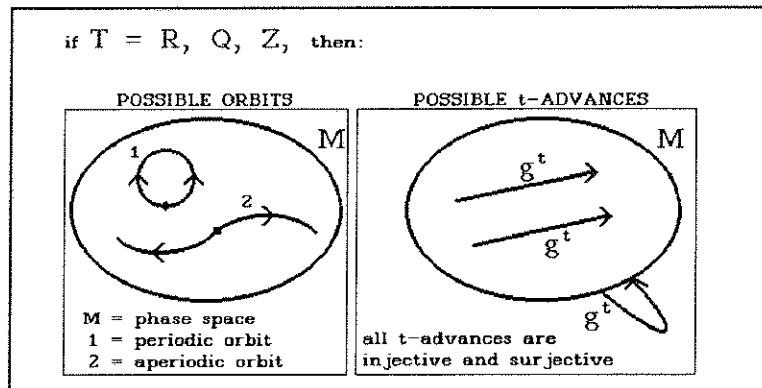


Figure 4 Orbits and t -advances in reversible systems

- (2) for any $t \in T$, g^t is a bijection; therefore, the inverse function of g^t , $(g^t)^{-1}$, is also a function from M to M , and $(g^t)^{-1} = g^{-t}$;
- (3) for any $x \in M$, there is exactly one orbit which passes through x ; for any $t \in T$, $x \in M$, there is exactly one state evolution g^z such that $g^z(t) = x$ (in fact $z = g^{-t}(x)$).

If $T = \mathbb{Z}^+, \mathbb{Q}^+, \text{ or } \mathbb{R}^+$, then:

(4) $\{g^t\}$ is a commutative monoid with respect to the composition operation \circ , and the unity is g^0 ;

(5) for any $x, y, z \in M$, for any $t, w \in T$, if $g^t(x) = g^w(y) = z$, then $\text{orb}(z) \subseteq \text{orb}(x)$ and $\text{orb}(z) \subseteq \text{orb}(y)$ ¹⁴.

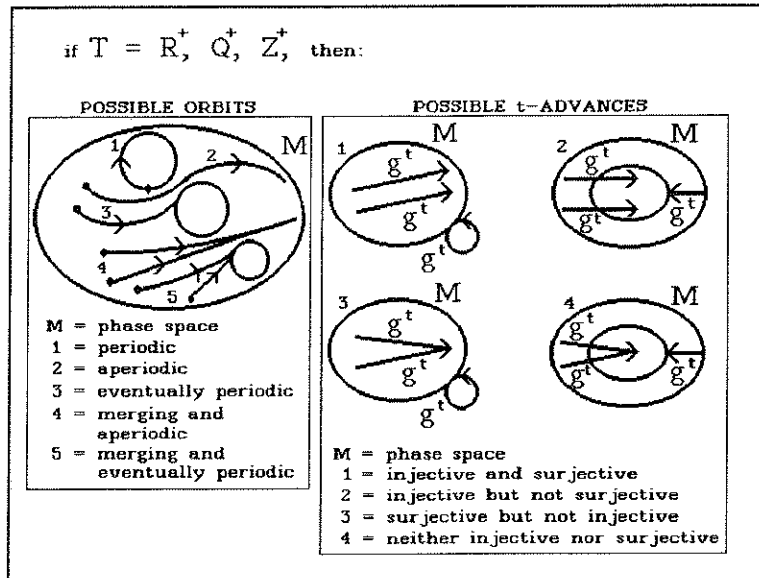


Figure 5 Orbits and t-advances in irreversible systems

If S is a deterministic dynamical system and $T = \mathbb{Z}$ or \mathbb{Z}^+ , then $g(t, x)$ is generated recursively by the two t-advances g^0 and g^1 . In fact:

$$\begin{aligned} g(0, x) &= g^0(x) = x \\ g(t+1, x) &= g^1(g(t, x)) \end{aligned}$$

If $T = \mathbb{Z}$, g^1 is injective and surjective (by property (2)). For negative instants, the following equality holds:

$$g(t-1, x) = (g^1)^{-1}(g(t, x)) \quad \text{where } (g^1)^{-1} = g^{-1} \text{ (by property (2)) is the inverse of } g^1.$$

Conversely, if $T = \mathbb{Z}$ or \mathbb{Z}^+ , and a function $G: M \rightarrow M$ is specified, we can define recursively $g(t, x)$ by first stipulating $g^1 =: G$, and $g^0 =: \text{identity function on } M$.

¹⁴ This property expresses the fact that in *irreversible systems* (that is, systems where time is limited to the non-negative portion of \mathbb{R} , \mathbb{Q} , or \mathbb{Z}) there may be merging orbits, but no intersecting orbits.

$\text{If } T = Z^+ : g^k(x) = \underbrace{g^1(g^1(\dots g^1(x)\dots))}_{k \text{ times}}$
$\text{If } T = Z : g^k(x) = \underbrace{g^1(g^1(\dots g^1(x)\dots))}_{k \text{ times}}$ <p style="text-align: center;">and</p> $g^{-k}(x) = \underbrace{g^{-1}(g^{-1}(\dots g^{-1}(x)\dots))}_{k \text{ times}}$

If $T = Z$, G must be injective and surjective.

It is easy to verify that $\langle T M g(t x) \rangle$, where g is defined by the equations above, is a deterministic

Figure 6 Cascades

dynamical system. A deterministic system which is generated in this way by $G: M \rightarrow M$ is called a cascade on M and it can thus be simply indicated by $\langle T M G \rangle$, where $T = Z$ or Z^+ , and G is injective and surjective if $T = Z$.

Example 1.2 (iteration of the logistic function)

A family of simple cascades on \mathbb{R} which leads to an extremely complex dynamical behavior is the one generated by the logistic function. Here $T = Z^+$, $M = \mathbb{R}$, and G is defined by $G(x) = ax(1 - x)$, where a is a parameter. For example, the cascade determined by the value $a = 4$ displays chaotic behavior. If $0 < x < 0.5$ or $0.5 < x < 1$, the orbit of x almost completely fills up the unit interval¹⁵.

¹⁵ Kocak (1986, 23). For a complete analysis of the dynamical behavior of the logistic function see Devaney (1989).

3. Isomorphic systems and computational systems

An important problem concerning dynamical systems consists in determining whether two systems are equivalent from the point of view of their dynamical behavior. Intuitively, this will be the case if there is a one to

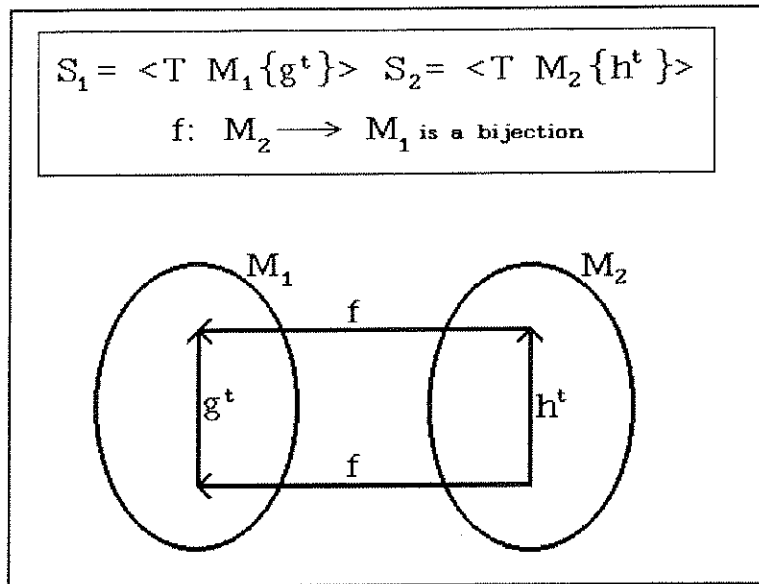


Figure 7 Isomorphic systems

one mapping between the states of the two systems so that each state evolution of one system will correspond, through this mapping, to a state evolution of the other (and conversely). We can make this idea precise as follows:

Definition 2 (isomorphic dynamical systems)

S_1 is isomorphic to S_2 iff:

- (1) $S_1 = \langle T, M_1, g(t, x) \rangle$ and $S_2 = \langle T, M_2, h(t, x) \rangle$
- (2) there is $f: M_1 \rightarrow M_2$ such that f is a bijection and, for any $t \in T$, for any $x \in M_1$, $f(g^t(x)) = h^t(f(x))$.

It is easy to verify that the isomorphism just defined is in fact an equivalence relation on the set of all deterministic dynamical systems. This set is thus divided into equivalence classes, and any two systems in each of these classes can be taken to be the same system as far as their dynamical behavior is concerned.

Theorem 1

The relation of isomorphism between dynamical systems is an equivalence relation

proof:

we must prove that the relation of isomorphism between dynamical systems is reflexive, symmetric, and transitive. These three properties follow immediately from definition 2. For the details see the appendix q.e.d.

The main thesis of this chapter is that all computational systems are a special type of dynamical systems, and that recognizing this fact allows us to better understand many important properties of computational systems. As a first confirmation of this thesis, I show that any Turing machine can naturally be identified with two different cascades which, however, turn out to be isomorphic.

The future behavior of a Turing machine is determined once the position x of the head, the internal state q , and the content of the tape p are specified. Any such a triple $\langle x q p \rangle$ can thus be identified with a *total state* of the Turing machine. For technical reasons, however, it is often convenient to identify a total state with the finite string of symbols $vqaz$, where q is the internal state of the Turing machine, a is the symbol on which the reading head is positioned, v is the string of symbols to the left of a up to the leftmost non-blank symbol (included), and z is the string of symbols to the right of a up to the rightmost non-blank symbol (included); v (or z) is empty if the leftmost (or rightmost) symbol is not to the left (or to the right) of a . I show below that these two natural representations of the total states of a Turing machine are in fact equivalent. This means that the two different cascades which correspond to the two representations are isomorphic

dynamical systems.

Example 2.1 (two isomorphic dynamical systems which individuate the same Turing machine)

An arbitrary Turing machine can be naturally identified with at least two different dynamical systems which are, intuitively, the same system. The two systems which correspond to the same Turing machine turn out to be isomorphic in the sense of definition 2.

Let C be an arbitrary Turing machine. Let $A = \{a_j\}$ be the alphabet of C , where A contains m symbols ($0 < m \in \mathbb{Z}^+$). One of the symbols in A (say a_0) is called the blank and is indicated by "b". Let $Q = \{q_r\}$ be the set of all internal states of C , where Q contains k states ($0 < k \in \mathbb{Z}^+$). A quadruple is any string of the form $q_r a_j D q_s$, where $D = a_j, L, \text{ or } R$, "L" stands for "left", and "R" stands for "right". An arbitrary Turing machine C is specified by a consistent set G of km quadruples¹⁶, where k is the number of internal states in Q , m is the number of symbols in A , and two quadruples are consistent just in case they are identical or they do not begin with the same pair state/symbol. Consider now $W_1 =$: the set of all finite strings $v q_r a_j z$, where a_j is an arbitrary symbol in A , q_r is an arbitrary internal state in Q , v and z are finite strings of symbols in A (v or z may be empty). The Turing machine C is supposed to be in internal state q_r and read the symbol a_j immediately to the right of q_r . Each string in W_1 is thus a total state of C because, given the set of quadruples G , and $w \in W_1$, the complete future behavior of C is determined. In fact, G determines a function¹⁷ $g[G]_1: W_1 \rightarrow W_1$ which is defined as follows. Let $q_r a_j D q_s$ be the quadruple in G which begins with $q_r a_j$. If $D = a_i$, $g[G]_1(v q_r a_j z) = v q_s a_i z$; if $D = L$ and v is empty, $g[G]_1(v q_r a_j z) = v q_s b z'$, where $z' = a_j z$; if $D = L$ and $v = v' a_i$, $g[G]_1(v q_r a_j z) = v' q_s a_i z'$, where $z' = a_j z$; if $D = R$ and z is empty, $g[G]_1(v q_r a_j z) = v' q_s b z$, where $v' = v a_j$; if $D = R$ and $z = a_i z'$, $g[G]_1(v q_r a_j z) = v' q_s a_i z'$, where $v' = v a_j$. We can then consider the cascade $S_1 = \langle \mathbb{Z}^+ W_1 g[G]_1 \rangle$, and we can thus identify C with $S_1 = \langle \mathbb{Z}^+ W_1 g[G]_1 \rangle$.

On the other hand, it is also clear that we could identify an arbitrary total state of C with a triple $\langle x \ q \ p \rangle$, where x is a position of the machine's head, q is an internal state, and p is a finite string of symbols in A surrounded by an infinite number of blanks (that is, p is the content of an arbitrary tape). Let W_2 be the set of all such

¹⁶ This definition of a Turing machine is apparently more restrictive of the usual one which allows any consistent set of quadruples, but in fact it is not. If such a set does not contain a quadruple beginning with $q_r a_j$, we simply add the quadruple $q_r a_j q_r$, and we adopt a slightly different convention to determine when a Turing machine stops. A Turing machine stops iff its head position, its internal state, and all the symbols on the tape no longer change. This is equivalent to requiring that the Turing machine's total state no longer change (see below for the definition of a total state).

¹⁷ If G is a consistent set of km quadruples, $g[G]_1$ is a function defined for all total states of C ; if G were allowed to be any consistent set of quadruples, $g[G]_1$ might be undefined for some total state w .

triples. Then G determines a function $g[G]_2: W_2 \rightarrow W_2$, and C can also be identified with the cascade $S_2 = \langle Z^+ W_2 g[G]_2 \rangle$.

The two dynamical systems S_1 and S_2 , however, turn out to be isomorphic in the sense of definition 2. This is clear when the obvious one to one correspondence between W_1 and W_2 is considered.

A second fact about Turing machines which is usually accepted without proof is that changing the alphabet, or switching the direction of motion, does not matter. In other words, if a machine C^* can be obtained by relabeling the symbols and the states of another machine C , and by switching Left and Right, then C^* is essentially the same machine as C . This can be formally proved once the cascades which correspond to C and C^* are considered. In fact, these two cascades are isomorphic in the sense of definition 2.

Example 2.2 (changing the alphabet of a Turing machine, and switching Left and Right, generates an isomorphic machine)

If C is an arbitrary Turing machine, let C^* be a second Turing machine obtained by relabeling the symbols and the states of C , and by switching L and R . It is intuitively clear that C and C^* are essentially the same machine. This can be formally proved once the dynamical systems which correspond to C and C^* are considered. These two dynamical systems turn out to be isomorphic in the sense of definition 2.

Let A and A^* be, respectively, the alphabets of C and C^* , where $h: A \rightarrow A^*$ is a bijection. Let Q and Q^* be, respectively, the state-sets of C and C^* , where $f: Q \rightarrow Q^*$ is a bijection. If $q_i a_j a_k q_l$ is a quadruple of C , $f(q_i)h(a_j)h(a_k)f(q_l)$ is a quadruple of C^* ; if $q_i a_j R q_l$ is a quadruple of C , $f(q_i)h(a_j)Lf(q_l)$ is a quadruple of C^* ; if $q_i a_j L q_l$ is a quadruple of C , $f(q_i)h(a_j)Rf(q_l)$ is a quadruple of C^* ; nothing else is a quadruple of C^* . C can be identified with $S = \langle Z^+ W g[G] \rangle$, where $W =$: the set of all finite strings $vq_i a_j z$ such that a_j is an arbitrary symbol in A , q_i is an arbitrary internal state in Q , v and z are finite (possibly empty) strings of symbols in A , and the Turing machine C is supposed to be in internal state q_i and read the symbol a_j immediately to the right of q_i . Similarly, I identify C^* with $S^* = \langle Z^+ W^* g[G^*] \rangle$, where $W^* =$: the set of all finite strings $u s_j p_i r$ such that s_j is an arbitrary symbol in A^* , p_i is an arbitrary internal state in Q^* , u and r are finite (possibly empty) strings of symbols in A^* , and the Turing machine C^* is supposed to be in internal state p_i and read the symbol s_j immediately

to the left of p_i . For any finite string w^* of symbols in $A^* \cup Q^*$, $c(w^*)$ is the finite string obtained by reversing the order of all the symbols in w^* . Let $\rho: W \rightarrow W^*$ satisfy: $\rho(vq_i a_j z) = c(h(v) f(q_i) h(a_j) h(z))$, where $h(v)$ (or $h(z)$) is the string obtained by applying h to each symbol in v (or in z) and by concatenating the results. By its definition, ρ is a bijection between W and W^* , and it is immediate to verify that ρ satisfies definition 2. S and S^* are thus isomorphic dynamical systems.

We are now in the position of defining a proper subset of the deterministic dynamical systems which can be identified with the class of all computational systems. *Computational systems have two essential features. First, they are deterministic systems which evolve in discrete time steps. Second, they can always be described in an effective manner.*

The first condition can be made precise by requiring that any computational system be a *cascade*. I propose to express the second condition by means of the relation of *isomorphism* between dynamical systems. The intuitive idea is that a cascade $S = \langle T M G \rangle$ is describable in an effective manner just in case S is isomorphic to a second cascade $S_1 = \langle T M_1 H \rangle$ such that (i) there is an effective procedure for determining the possible states of its phase space M_1 ; (ii) its transition function H is effective.

Turing's analysis of an effective procedure is usually accepted (Turing 1965, sec. 9). According to this view, any effective procedure for transforming finite symbol structures can be reduced to a *function which can be computed by a Turing machine*¹⁸. If we accept Turing's thesis, we can thus define a

¹⁸ I will take up this issue again in chapter 2, where I will propose a more general analysis of the intuitive concept of an effective transformation of finite symbol structures. Ordinary Turing machines operate on finite symbol structures (strings) written on an infinite support (a linear tape)

computational system as a cascade $S = \langle T M G \rangle$ isomorphic to a second cascade $S_1 = \langle T M_1 H \rangle$ whose phase space M_1 is decidable¹⁹, and whose transition function H is computable by some Turing machine. That is:

Definition 3
(computational system)

S is a computational system iff:

S is a cascade and there is S_1 such that:

- (1) $S_1 = \langle T M_1 H \rangle$ is a cascade;
- (2) S is isomorphic to S_1 ;
- (3) if $P(A)$ is the set of all finite strings built out of some finite alphabet A , $M_1 \subseteq P(A)$ and there is a Turing machine which computes²⁰ the characteristic function

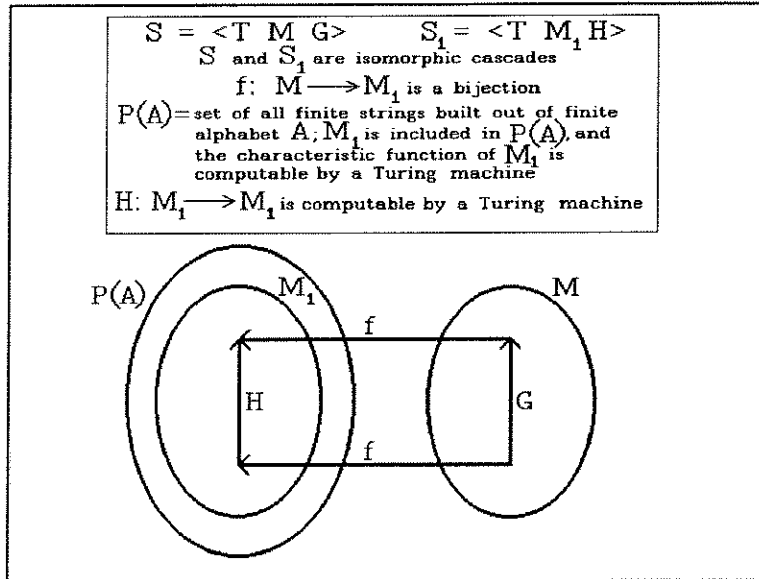


Figure 8 Computational systems

whose topography is extremely simple. But one can think of effectively transforming finite symbol structures written on an infinite support whose topography is arbitrarily complicated. It is in fact possible to define *generalized Turing machines* which operate on these 'supertapes' in exactly the same way as ordinary Turing machines operate on linear tapes. The interesting result is that some of these generalized Turing machines are able to compute non-recursive functions. At the end of chapter 2, I will propose a more comprehensive definition of a computational system (a *generalized computational system on a regular pattern field*, ch. 2, sec. 5, def. 8), which is based on the concept of a generalized Turing machine. The definition of a *computational system* which I give below (def. 3) turns out to be a special case of the one I give in chapter 2. In fact, generalized computational systems on a regular pattern field reduce to computational systems when the pattern field is identified with the tape of an ordinary Turing machine.

¹⁹ This means that the characteristic function of M_1 is computable by some Turing machine.

²⁰ A Turing machine C computes a function $f: X \rightarrow Y$ (where $X, Y \subseteq P(A)$) just in case $\$$ and $\#$ are two symbols which do not belong to the alphabet A and, for any $x \in X$, when C is started in initial state q_0 , on the left marker $\$$ of a tape whose content is $\dots bb\$x\#bb\dots$, C stops and the content of the final tape is $\dots bb\$f(x)\#bb\dots$

The characteristic function of $M_1 \subseteq P(A)$ is $c: P(A) \rightarrow \{b, bb\}$, where b is the string which only contains one blank, bb is the string obtained by concatenating two blanks, and $c(x) =: b$ if $x \in M_1$, $c(x) =: bb$ if $x \notin M_1$.

- of M_1 ;
- (4) there is a Turing machine which computes H and, if $T = Z$, there is also a Turing machine which computes H^{-1} .

If definition 3 does indeed capture the intuitive concept of a computational system²¹, any system which is usually studied by computation theory must fall within its scope. In particular, Turing machines themselves must turn out to be computational systems. This is in fact the case, as the following example shows.

Example 3.1 (all Turing machines are computational systems)

Let C be an arbitrary Turing machine with alphabet A , state-set Q , and quadruple-set G . As we have seen in example 2.1, C can be identified with $S = \langle Z^+ W g[G] \rangle$, where $W =$: the set of all finite strings $uqaz$ such that a is an arbitrary symbol in A , q is an arbitrary internal state in Q , u and z are finite (possibly empty) strings of symbols in A , and the machine is supposed to be in internal state q and read the symbol a immediately to the right of q . This shows that, for an arbitrary Turing machine C , C is a cascade. Conditions (1) and (2) of definition 3 are also obviously satisfied for $C = S = S_1$. Condition (3) holds because W is a proper subset of the set $P(A \cup Q)$ of all the finite strings built out of the alphabet $A \cup Q$, and it is obvious that there is a Turing machine C_1 which computes the characteristic function of W . Finally, condition (4) is also satisfied because we can construct a second Turing machine C_2 which computes the transition function $g[G]$ of the Turing machine C . The alphabet of C_2 is $A \cup Q \cup \{ \$ \}$, and C_2 operates as follows. C_2 is started on the left marker $\$$ of a tape whose content is $\dots bb\$uqaz\#bb\dots$, and it goes to the right until it encounters q . Then, C_2 operates the transformation which corresponds to the quadruple of C which begins with qa , taking care of repositioning the left marker $\$$ or the right marker $\#$ if needed. After this, C_2 stops. Also notice that this argument does not depend on the particular cascade I I have chosen to represent C . If C were identified with any other cascade isomorphic to S , definition 3 would still be satisfied.

²¹ Notice that *being a cascade* corresponds to the property of being a deterministic system which evolves in discrete time steps. On the other hand, *being isomorphic to a second cascade whose phase space is decidable and whose transition function is Turing computable* expresses the fact that a computational system can always be described in an effective manner.

Example 3.1 also shows that there is nothing wrong with the apparent circularity of definition 3. The general concept of a computational system is defined by means of the notion of a Turing machine, which obviously is a special kind of computational system. Therefore, in a certain sense, the notion of a computational system is presupposed, and one might worry that this circularity invalidate definition 3. But this is not a real problem. In the first place, the circularity of definition 3 is not complete, because the general concept of a computational system is defined by means of one of its specifications, but it is not *directly* defined in terms of itself. In the second place, the partial circularity of definition 3 would be vicious if the specific concept I have used to define the general one could not be *proved* to be a special case of the defined concept. But example 3.1 does in fact provide this proof.

A complete proof of the adequacy of definition 3 is out of the question, for that would involve showing that any *possible* formal specification of the concept of a computational system is a special case of definition 3. Obviously, we cannot know all the possible specifications of this concept, so that such a proof cannot be produced. It is not however difficult to show that all the *known* formal definitions of specific computational systems (register machines, cellular automata, monogenic production systems, etc.) satisfy definition 3. Intuitively this is to be expected, for each of these systems can be emulated by a suitably designed Turing machine. The next example outlines the proof for linear cellular automata. Analogous arguments can then be given for all other cases.

Example 3.2 (all linear cellular automata are computational systems)

A linear cellular automaton is constituted by a doubly infinite sequence of cells. At any time, each cell is in one of n possible states. The next state of each cell is determined by its present state and by the present state of k cells to its left and k cells to its right. Updating is synchronous. Each cellular automaton C is thus specified by a rule $\phi: A^{2k+1} \rightarrow A$, where A is the set of all possible cell states. One of the n states is called the blank state, and is conventionally indicated by b . I also require that the function ϕ take the sequence of $2k+1$ b s to b , and that, at any time, only a finite number of cells be in a non-blank state²². A cell is *quiescent* just in case all the cells in its neighborhood are in the blank state. I now show that all linear cellular automata satisfy definition 3.

Let C be an arbitrary $\langle n, k \rangle$ linear cellular automaton²³, and let ϕ be its rule. A total state of C can be identified with the finite sequence of cell states between the leftmost and the rightmost non-quiescent states (included). I identify the total state whose cells are all quiescent with the finite sequence which only contains one blank. Let W be the set of all total states of C . The rule ϕ obviously determines a function $\Phi: W \rightarrow W$, and C can thus be identified with $S = \langle Z^+ W \Phi \rangle$. This shows that C is a cascade. Conditions (1) and (2) of definition 3 are also obviously satisfied for $S_1 = S = C$. Condition (3) also holds because $w \in W$ just in case either (i) w only contains one blank, or (ii) $w = b \dots b a_L u a_R b \dots b$, where a_L and a_R are, respectively, the symbols on the leftmost and rightmost non-blank cells, u is the string of symbols between a_L and a_R (u may be empty and, if this is the case, a_L may be equal to a_R), and $b \dots b$ is a string of k blanks. Therefore, $W \subseteq P(A)$, and it is obvious that there is a Turing machine which computes the characteristic function of W .

I finally show that the transition function Φ of S is computable by a Turing machine. This Turing machine starts on the left marker $\$$ of a tape $\dots bb\$w\#bb\dots$ whose content represents an arbitrary total state $w \in W$ of the cellular automaton. Each square of the tape can thus be identified with the corresponding cell of the cellular automaton. The Turing machine then goes to the right, sequentially updating all the cells up to the right marker $\#$, according to rule ϕ . When the new state a_j of a cell is computed, the Turing machine provisionally assigns to that cell the symbol $a_i a_j$ which codes for both the present state a_i and the new state a_j . This allows for the correct updating of the next cell. When all the new states have been computed, the Turing machine first repositions the right marker $\#$ (if needed) and then goes back, replacing each provisional symbol $a_i a_j$ with the updated state a_j of each cell. The Turing machine ends this routine as soon as it encounters the left marker $\$$, it then repositions this marker (if needed), and it stops. I have thus shown that C is a computational system.

²² If either condition is not satisfied, an infinite number of cells may need to be updated in one time step. No Turing machine could thus compute this step in a finite time. Cellular automata which do not satisfy these conditions may thus be more powerful than Turing machines, and they may not be computational systems in the sense of definition 3.

²³ C is a $\langle n, k \rangle$ linear cellular automaton iff: each cell has n possible states and the updating of each cell is determined by a neighborhood of radius k .

4. Emulation of a system

The fact that a certain system can be emulated by a different system is familiar to any student of computation theory. The relation which holds between any two such systems can be

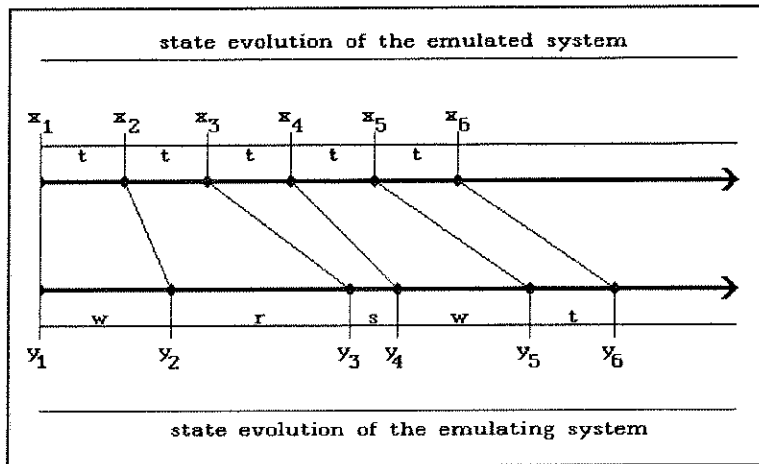


Figure 9 Two state evolutions of two systems in the emulation relation

intuitively characterized as follows: some state evolutions of the emulating system can be divided into consecutive parts so that the two states reached by this system at the end of two consecutive parts correspond to one step in the evolution of the emulated system. The emulation relation is not peculiar to computational systems, but it may in fact hold between two arbitrary deterministic dynamical systems.

This concept can be made precise by considering two functions, u and v , which relate the states and the t -advances of the two systems. The function u injectively maps the states of the emulated system into the states of the emulating one. Therefore, under this mapping, each state of the emulated system can be identified with a corresponding state of the emulating one. The function v , instead, maps each step of the emulated system into a corresponding step of the emulating one. Each step is individuated by the state involved in that step and by the length

of the step, that is, by the time necessary for transforming the state into another state. The function v tells us the length of the corresponding step of the emulating system. For instance, suppose we consider state x and t -advance g^t . The function v , when applied to x

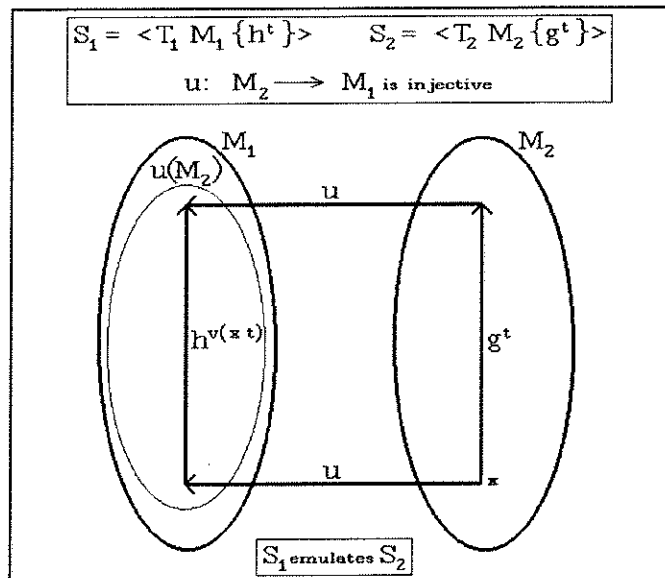


Figure 10 Emulation of a system

and t , gives us the time $v(x, t)$ which the emulating system will employ to transform the state $u(x)$ which corresponds to x into the state $u(g^t(x))$ which corresponds to $g^t(x)$. Therefore, the functions u and v must satisfy: $u(g^t(x)) = h^{v(x,t)}(u(x))$, where $h^{v(x,t)}$ is the t -advance of the emulating system which corresponds to step $g^t(x)$. Obviously, we should also require that v assign longer times to longer steps, and that a step of length zero correspond to time zero. Finally, since each state in the image of u is identified with a state of the emulated system, we should require that a transformation of any two such states correspond to a step of the emulated system. That is, if $h^t(u(x)) = u(y)$, there must be a time w such that $g^w(x) = y$ and $v(x, w) = t$. I give below the formal definition of the emulation relation:

Definition 4 (emulation)

S_1 emulates S_2 iff:

- (1) $S_1 = \langle T_1 M_1 h(t, x) \rangle$ and $S_2 = \langle T_2 M_2 g(t, x) \rangle$;

(2) there are u and v such that:

(a) $u: M_2 \rightarrow M_1$ and u is injective;

(b) let $T_{i+} =: \{t: t \in T_i \text{ and } t \geq 0\}$; $v: M_2 \times T_{2+} \rightarrow T_{1+}$ such that:

$v(x, 0) = 0$; if $w < t$, then $v(x, w) < v(x, t)$;

(c) for all $t \in T_{2+}$, $u(g^t(x)) = h^{v(x,t)}(u(x))$;

(d) for all $t \in T_{1+}$, if $h^t(u(x)) = u(y)$, there is a time w such that $g^w(x) = y$ and $v(x, w) = t$.

It follows from conditions (2b) and (2c) of definition 4, that the function v also satisfies:

$v(x, w) = 0$ iff $w = 0$

$v(x, w) > 0$ iff $w > 0$

$v(x, w+z) = v(x, w) + v(g^w(x), z)$.

The function u maps the states of the emulated system into states of the emulating system which thus represent them. Intuitively, the meaning of the function v is the following: $v(x, t)$ is the time which the emulating system employs to emulate the step from x to $g^t(x)$ of the emulated system. This time may depend on 'how long' such a step is (that is, on t) and also on the particular state x which is transformed by such a step. Conditions (2b) ensure that the dependence of v on time is correct, in the sense that a 'zero step' (x transformed into x) is emulated in zero time, and longer steps are emulated in longer times. Condition (2c) ensures that any step of the emulated system is emulated by the emulating system. Conversely, condition (2d) ensures that any step of the emulating system which transforms states which represent states of the emulated system does in fact emulate a corresponding step of this system.

The reason for restricting conditions (2b), (2c), and (2d) to positive instants is

that I want definition 4 to apply to 'mixed cases', that is, cases in which one of the systems only evolves through positive times, while the other one may also perform negative steps. If the restriction to positive instants were dropped, an arbitrary mixed case would not satisfy the emulation relation.

Example 4.1 (a simple cellular automaton which emulates another one)

Stephen Wolfram has noticed that there are very simple cellular automata which emulate other cellular automata (Wolfram 1983a, 629-30). Let C_0 and C_1 be linear cellular automata with cell states $\{0, 1\}$. The next state of each cell depends on its present state and on the present states of its left and right neighbors. These two automata are individuated by the following rules:

rules 22 and 146 (in binary)

C_0	1	1	1	1	1	0	1	0	1	1	0	0	1	0	0	0			
	0		0		0		1		0		1		1		0		(rule 22)		
C_1	1	1	1	1	1	0	1	0	1	1	0	1	0	0	0	1	0	0	0
	1		0		0		1		0		0		1		0				(rule 146)

It turns out that C_0 emulates C_1 in the sense of definition 4. Let W be the set of all doubly infinite sequences of 0s and 1s which contain a finite number of 1s. W is thus the phase space of both C_0 and C_1 . To see that C_0 emulates C_1 , let $u: W \rightarrow W$ be the function which, for an arbitrary $w \in W$, replaces 0 with 00, and 1 with 01. Let $v: W \times \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$ be defined by: $v(w, k) = k$, if w does not contain any 1; $2k$, otherwise. It is then easy to verify that u and v satisfy definition 4 (substitute 00 for 0 and 01 for 1 in each transformation of rule 146, and then apply rule 22).

Example 4.2 (linear cellular automata emulate Turing machines)

For an arbitrary Turing machine C_1 , there is a linear cellular automaton C_0 such that C_0 emulates C_1 . C_0 is a linear cellular automaton with a neighborhood of radius 3, and C_0 is constructed as follows²⁴.

²⁴ This construction is simple and extremely natural, but it is not the most economical one. If C_1 has m symbols and n internal states, C_0 has $m+n$ states. It is possible to reduce this number to $\max(m+1, n+1)$ (Smith 1971, th. 3).

If A and Q are respectively, the alphabet and the state-set of the Turing machine C_1 (where $A \cap Q = \emptyset$), $A \cup Q$ is the state set of the cellular automaton C_0 . If b is the blank of C_1 , b is the blank state of C_0 . If $\dots a a a q a a a \dots$ is an arbitrary total state of C_1 , $\dots b a b a q a b a b a \dots$ is the total state of C_0 which represents $\dots a a a q a a a \dots$. Given the quadruples of C_1 , the rule of C_0 is obtained as follows. An arbitrary quadruple of C_1 has three possible forms: $q_i a_j a_k q_l$, $q_i a_j R q_l$, $q_i a_j L q_l$. Each of these quadruples can be rewritten as a matrix of two rows and eleven columns, where a is a place-holder²⁵ for an arbitrary symbol in A , and b is the blank state of C_0 :

	1	2	3	4	5	6	7	8	9	10	11	
1	a	b	a	b	a	q_i	a_j	b	a	b	a	corresponds to $q_i a_j a_k q_l$
2						q_l	a_k					

	1	2	3	4	5	6	7	8	9	10	11	
1	a	b	a	b	a	q_i	a_j	b	a	b	a	corresponds to $q_i a_j R q_l$
2						b		q_l				

	1	2	3	4	5	6	7	8	9	10	11	
1	a	b	a	b	a	q_i	a_j	b	a	b	a	corresponds to $q_i a_j L q_l$
2				q_l		b						

Each of the quadruples may change the state of at most two cells: cells 6 and 7 for the first quadruple, cells 6 and 8 for the second, and cells 4 and 6 for the third. All other cells maintain their previous states. Each quadruple thus corresponds to two transformation-forms of the cellular automaton:

	a	b	a	q_i	a_j	b	a					
				q_l								correspond to $q_i a_j a_k q_l$
	b	a	q_i	a_j	b	a	b					
				a_k								

²⁵ This means that different copies of a may represent different symbols of the alphabet A . Therefore, a is not a variable.

$ \begin{array}{cccccc} a & b & a & q_i & a_j & b & a \\ & & & b & & & \end{array} $	correspond to $q_i a_j R q_i$
$ \begin{array}{cccccc} a & q_i & a_j & b & a & b & a \\ & & & q_i & & & \end{array} $	

$ \begin{array}{cccccc} a & b & a & b & a & q_i & a_j \\ & & & q_i & & & \end{array} $	correspond to $q_i a_j L q_i$
$ \begin{array}{cccccc} a & b & a & q_i & a_j & b & a \\ & & & b & & & \end{array} $	

The cellular automaton rule which individuates C_0 is obtained from these transformation-forms by substituting all possible symbols in A for each occurrence of a . Any sequence of seven cell states which cannot be obtained from these transformation-forms is taken to the state of its central cell. Let u and v be defined by: $u(\dots a a q a a a \dots) =: \dots b a b a b a q a b a b a \dots$; $v(\dots a a a q a a a \dots k) =: k$. Then, by construction of C_0 , and by the definitions of u and v , C_0 emulates C_1 .

The emulation relation is reflexive and transitive, but it is not symmetric. It is thus a quasi-ordering on the set of all deterministic dynamical systems. The proof that reflexivity holds is immediate, once u is taken to be the identity function and v the function which takes $\langle x \ t \rangle$ to t . As for transitivity, suppose S_1 emulates S_2 , S_2 emulates S_3 , and that $u_1: M_2 \rightarrow M_1$, $u_2: M_3 \rightarrow M_2$, $v_1: M_2 \times T_{2+} \rightarrow T_{1+}$, $v_2: M_3 \times T_{3+} \rightarrow T_{2+}$ satisfy definition 2. Then it is easy to verify that $u_3 =: u_1(u_2(z))$ and $v_3(z \ t) =: v_1(u_2(z) \ v_2(z \ t))$ satisfy definition 2. I give the details of the proof in the appendix.

Theorem 2

The emulation relation is a quasi-ordering on the set of all deterministic dynamical systems

proof:

see the appendix

5. Reversible versus irreversible systems

Let T be the reals, the rationals, or the integers, and let T^+ be the non-negative reals, non-negative rationals, or non-negative integers. Then, S is a reversible system just in case $S = \langle T \ M \ g(t \ x) \rangle$, and S is irreversible if, and only if, $S = \langle T^+ \ M \ g(t \ x) \rangle$. We have seen above (section 2, properties 1-3) that, for any reversible system, $\{g^t\}$ is a group with respect to the composition operation and that the inverse of $g^t = (g^t)^{-1} = g^{-t}$. Therefore, if x is the state of the system at the

Deterministic dynamical systems			
Reversible		Irreversible	
$T = \mathbb{R}, \mathbb{Q}, \mathbb{Z}$		$T = \mathbb{R}^+, \mathbb{Q}^+, \mathbb{Z}^+$	
Not time symmetric	Time symmetric There is $-\cdot: M \rightarrow M$: $-(g^t(-x)) = g^{-t}(x)$	Quasi-reversible For any g^t , $x \longrightarrow y$	Not quasi-reversible There is g^t such that: $x \longrightarrow y$
			Weakly irreversible Irreversible, not quasi-reversible, and not strongly irreversible

Figure 11 Types of dynamical systems

present time t_0 , we can always find the state of the system at any previous time t_0-t by simply considering the negative t -advance $g^{-t}(x)$. If a system is irreversible this is not possible, because no negative t -advance is in $\{g^t\}$. Yet, one might consider whether g^t is injective for any t . If this is the case, and x is the state of

the system at the present time t_0 , we can still find the state of the system at any previous time $t_0 - t \geq 0$ by considering the inverse of the t -advance g^t , that is $(g^t)^{-1}(x)$. Any irreversible system such that g^t is injective for any t will be called quasi-reversible²⁶.

Suppose now that S is irreversible, but not quasi-reversible. It is useful to further distinguish two cases: S is strongly irreversible just in case there are t -advances g^t and g^z and there are x and y such that $x \neq y$, $g^t(x) = g^z(y)$ and, for all times w , $g^w(x) \neq y$ and $g^w(y) \neq x$; S is weakly irreversible just in case S is irreversible, S is not quasi-reversible, and S is not strongly irreversible.

If a system S is reversible, then g^t is surjective for any t . In irreversible systems this property usually fails. Any system such that g^t is not surjective for some t will be called a contracting system.

I now prove two theorems which are direct consequences of the definition of strongly irreversible system. The first theorem gives a necessary and sufficient condition for a system to be strongly irreversible: a system is strongly irreversible just in case there are two state evolutions whose images intersect, but are not in the subset relation. The second theorem simply states that a strongly irreversible system is irreversible, and has some t -advance which is not injective (that is, it is not quasi-reversible).

²⁶ Bennett (1973) calls these systems "logically reversible".

Theorem 3

S is strongly irreversible iff there are two state-evolutions, g^x and g^y , such that $x \neq y$ and $\text{Im}(g^x) \not\subseteq \text{Im}(g^y)$ and $\text{Im}(g^y) \not\subseteq \text{Im}(g^x)$ and $\text{Im}(g^x) \cap \text{Im}(g^y) \neq \emptyset$

proof:

assume that S is strongly irreversible. Then, there are x and y such that $x \neq y$ and $\text{Im}(g^x) \cap \text{Im}(g^y) \neq \emptyset$. If $\text{Im}(g^x) \subseteq \text{Im}(g^y)$ or $\text{Im}(g^y) \subseteq \text{Im}(g^x)$, there is w such that $g^w(x) = y$ or $g^w(y) = x$. But there is no such w because S strongly irreversible. Conversely, suppose first that S is reversible. Then, since $\text{Im}(g^x) \cap \text{Im}(g^y) \neq \emptyset$, $\text{Im}(g^x) = \text{Im}(g^y)$, against the hypotheses. S is thus irreversible. From the hypotheses of the theorem, it follows that there are x, y, t and z such that $g^t(x) = g^z(y)$ and $x \neq y$. If there is $w \geq 0$ such that $g^w(x) = y$ or $g^w(y) = x$, then $\text{Im}(g^x) \subseteq \text{Im}(g^y)$ or $\text{Im}(g^y) \subseteq \text{Im}(g^x)$, against the hypotheses q.e.d.

Theorem 4

If S is strongly irreversible, S is irreversible and S is not quasi-reversible.

proof:

suppose that S is strongly irreversible and reversible. Then, from theorem 3, there are x and y, such that $x \neq y$ and $\text{Im}(g^x) \not\subseteq \text{Im}(g^y)$ and $\text{Im}(g^y) \not\subseteq \text{Im}(g^x)$ and $\text{Im}(g^x) \cap \text{Im}(g^y) \neq \emptyset$. But, since S is reversible, for all x and y, either $\text{Im}(g^x) = \text{Im}(g^y)$ or $\text{Im}(g^x) \cap \text{Im}(g^y) = \emptyset$.

Suppose now that S is strongly irreversible and quasi-reversible. Then, for all t, g^t is injective, and there are $g^t, g^z, x,$ and y such that $x \neq y, g^t(x) = g^z(y)$ and, for all times w, $g^w(x) \neq y$ and $g^w(y) \neq x$. If $t = z, g^t$ is not injective. Suppose $t > z$. Then, $g^z(g^{t-z}(x)) = g^{z+t-z}(x) = g^t(x) = g^z(y)$. But $g^{t-z}(x) \neq y$, hence g^z is not injective. Analogously if $t < z$ q.e.d.

It is useful to look at the previous definitions and theorems from the following point of view. Recall first that the orbit of a point x is the image of the state evolution g^x whose initial state is x. I indicate the orbit of x by "orb(x)". In reversible systems, for any two orbits orb(x) and orb(y), there are only two possible cases: $\text{orb}(x) = \text{orb}(y)$, or $\text{orb}(x) \cap \text{orb}(y) = \emptyset$, and the first case holds iff $x \in \text{orb}(y)$ or $y \in \text{orb}(x)$. In quasi-reversible systems or weakly irreversible systems four

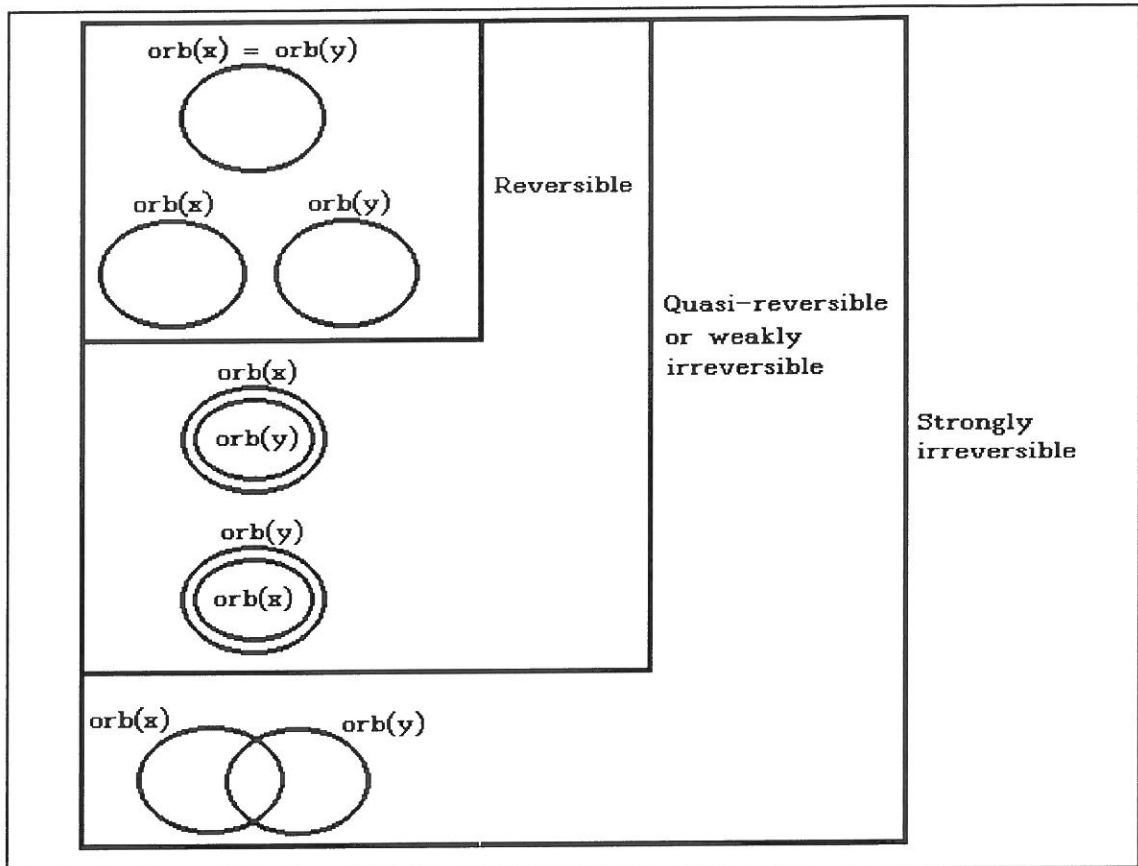


Figure 12 Set theoretical relationships between two orbits in different types of dynamical systems

cases are possible: $\text{orb}(x) = \text{orb}(y)$, or $\text{orb}(x) \cap \text{orb}(y) = \emptyset$, or $\text{orb}(x) \subset \text{orb}(y)$, or $\text{orb}(y) \subset \text{orb}(x)$. In strongly irreversible systems also a fifth case is possible: $\text{orb}(x) \not\subset \text{orb}(y)$, and $\text{orb}(y) \not\subset \text{orb}(x)$, and $\text{orb}(x) \cap \text{orb}(y) \neq \emptyset$; that is, the orbits of x and y intersect, but neither is a subset of the other. Hence, only in strongly irreversible systems are there orbits of different points which merge with time. I now define four types of different orbits, and I then study the relations between the classes of irreversible systems I have previously introduced and these four types of orbits.

Let, for any $x \in M$, $\text{orb}(x) =: \text{Im}(g^x)$. Then:
 $\text{orb}(x)$ is periodic iff: there is $t > 0$ such that $g^t(x) = x$;
 $\text{orb}(x)$ is eventually periodic iff: $\text{orb}(x)$ is not periodic, but there are y and t : $g^t(x) = y$ and $\text{orb}(y)$ is periodic;
 $\text{orb}(x)$ is aperiodic iff: $\text{orb}(x)$ is neither periodic, nor eventually periodic;
 $\text{orb}(x)$ is merging iff: there is y such that $x \neq y$, $\text{orb}(x) \cap \text{orb}(y) \neq \emptyset$, $\text{orb}(x) \not\subseteq \text{orb}(y)$, and $\text{orb}(y) \not\subseteq \text{orb}(x)$.

Any point (state) whose orbit is periodic, eventually periodic, aperiodic, or merging, is called a periodic, eventually periodic, aperiodic, or merging point.

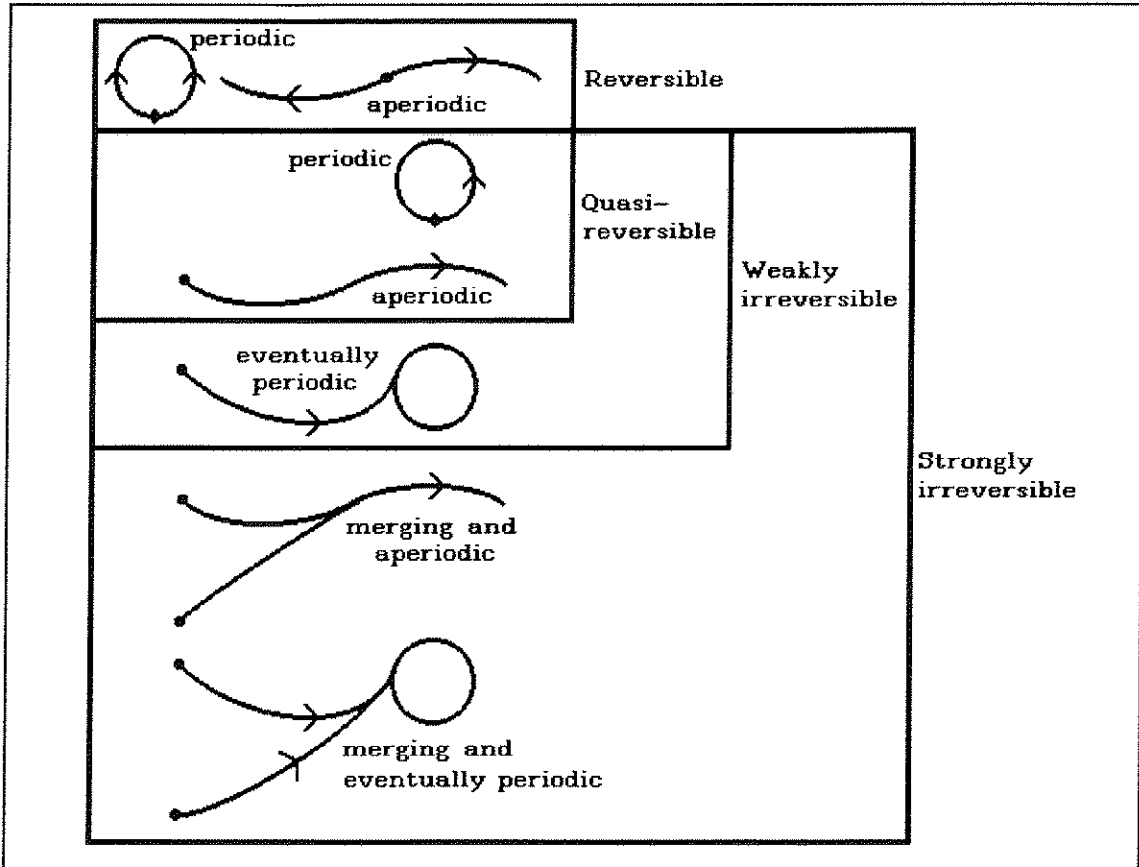


Figure 13 Possible orbits in different types of dynamical systems

In the first place, since the orbit of x is the image of the state evolution g^x , we can restate theorem 1 as follows:

Corollary 3.1

S is strongly irreversible iff S has merging orbits

proof:

from theorem 3 and the definition of merging orbit q.e.d.

Corollary 3.1 characterizes strongly irreversible systems in terms of a specific type of orbit they possess. A similar characterization of weakly irreversible systems is possible, but we first need two lemmas. The first lemma states that any system which has eventually periodic orbits or merging orbits has some t -advance which is not injective. If there are merging orbits, the thesis follows immediately from corollary 3.1 and theorem 4. As for the case of eventually periodic orbits, the thesis follows (with some fiddling) from the definitions of eventually periodic orbit and periodic orbit. The details of the proof are in the appendix.

Lemma 5.1

If S has eventually periodic orbits or S has merging orbits, there is t such that g^t is not injective

proof:

see the appendix

An interesting consequence of Lemma 5.1 is that any Turing machine which halts after at least one step has some t -advance which is not injective. The interesting point is that the halting problem for the class of all quasi-reversible

Turing machines is thus decidable. In fact, given any input, we can just check whether the machine immediately halts. If it does not, it will never halt for, otherwise, some t -advance would not be injective. What this means is that the unsolvability of the halting problem depends on the fact that some Turing machines are not quasi-reversible²⁷. We will see later that a second important property of Turing machines, universality, is necessarily associated with strong irreversibility.

The second lemma states that any system which does not have merging orbits, but has some t -advance which is not injective, has eventually periodic orbits. The hypotheses of the lemma, together with the definition of merging orbit, corollary 3.1, and the definition of strongly irreversible system, imply that there are $t, w, x, y,$ and z such that $x \neq y, g^t(x) = g^t(y) = z,$ and $g^w(x) = y.$ Then, by the definition of eventually periodic orbit, the lemma will be proved if we can show that $\text{orb}(z)$ is periodic and $\text{orb}(x)$ is not periodic. The proof that $\text{orb}(z)$ is periodic is immediate. The proof that $\text{orb}(x)$ is not periodic proceeds by reductio. We first assume that

²⁷ Also notice that the class of all quasi-reversible Turing machines is decidable. In fact, the t -advances of a Turing machine are injective iff the transition function is injective, and we can decide this condition by inspecting the quadruples. If W is the set of all total states of a Turing machine, each quadruple is an injective function from $X \subset W$ to $W,$ and the domains of any two quadruples are disjoint. Therefore, the transition function of a Turing machine is injective iff the images of any two quadruples do not overlap. Furthermore, the following conditions hold: (i) two quadruples $q_i a_i a_m q_n$ and $q_s a_s a_k q_l$ have overlapping images iff $m = k$ and $n = l;$ (ii) two quadruples $q_i a_i L q_n$ and $q_s a_s L q_l$ ($q_i a_i R q_n$ and $q_s a_s R q_l$) have overlapping images iff $n = l$ and $i = r;$ (iii) two quadruples $q_i a_i L q_n$ and $q_s a_s a_k q_l$ ($q_i a_i R q_n$ and $q_s a_s a_k q_l$) have overlapping images iff $n = l;$ (iv) two quadruples $q_i a_i L q_n$ and $q_s a_s R q_l$ have overlapping images iff $n = l.$ Obviously, these conditions allow us to decide whether there is a pair of quadruples with overlapping images. If no such a pair exists, the Turing machine is quasi-reversible, and its halting problem is thus decidable. For example, a simple quasi-reversible Turing machine is specified by the quadruples $G = \{q_0 b R q_0 q_0 11 q_1 q_1 b b q_1 q_1 1 R q_2 q_2 b R q_2 q_2 1 R q_0\}.$

orb(x) is periodic, that is, there is s such that $g^s(x) = x$, and we then consider the three cases $s = t$, $s > t$, and $s < t$. In each case, we deduce the contradiction $x = z$ and $x \neq z$. The details are in the appendix.

Lemma 5.2

If there is t such that g^t is not injective, and S has no merging orbits, S has eventually periodic orbits

proof:

see the appendix

We can now characterize weakly irreversible systems as those systems which have eventually periodic orbits but do not have merging orbits:

Theorem 5

S is weakly irreversible iff S has eventually periodic orbits and S has no merging orbits

proof:

from lemma 5.1, lemma 5.2, corollary 3.1, and the definition of weakly irreversible system q.e.d.

If some t-advance is not injective, that is, if a system is not quasi-reversible, there must be either eventually periodic or merging orbits, and also the converse statement holds:

Corollary 5.1

S has eventually periodic orbits or S has merging orbits iff there is t such that g^t is not injective

proof:

left/right is lemma 5.1. As for right/left, suppose there is t: g^t is not injective. Then, either S has merging orbits, or S has not merging orbits. From this and lemma 5.2 the thesis follows q.e.d.

Computational systems are usually strongly irreversible, while most systems of classical dynamics are reversible. In fact, these systems satisfy an even stronger property: they are time symmetric, in the following sense. A total state, x , of a system of classical dynamics is specified when all the positions and velocities are given. If we consider the state $-x$, obtained from x by changing the sign of all velocities, it holds: $-(g^t(-x)) = g^t(x)$, for all t and x. That is, the 'inverse' of the forward evolution from state $-x$ is equal to the backward evolution from state x . Obviously, if the previous property holds, $-(-x) = x$, for all x; in fact: $-(-x) = -(g^0(-x)) = g^0(x) = x$. This motivates the following definition: S is time symmetric iff: S is reversible and there is a function $-: M \rightarrow M$ such that, for all x and t, $-(g^t(-x)) = g^t(x)$.

Example

The Galilean model of free fall $S = \langle T YxV \{g^t\} \rangle$ is a time symmetric system. Recall that T is time, Y are the vertical positions of a freely falling body, V the vertical velocities, and $g^t: YxV \rightarrow YxV$ is defined by: $g^t(y v) = \langle (y + vt + 1/2ct^2) (v + ct) \rangle$. Let $-: YxV \rightarrow YxV$ be defined by $-(y v) = \langle y -v \rangle$. Then: $-(g^t(-y v)) = -(g^t(y -v)) = -((y -vt + 1/2ct^2) (-v + ct)) = \langle (y - vt + 1/2ct^2) (v - ct) \rangle = g^t(y v)$. S is thus time symmetric.

6. Realization of a system

We have seen in section 5 that strongly irreversible systems have merging orbits, while any two orbits of a reversible system either coincide or are separated. It is then intuitively clear that reversible systems cannot emulate strongly irreversible ones. This is in fact a special case of a more general result: no system which is not strongly irreversible can emulate a strongly irreversible system. The proof of this theorem is by reductio. We assume first that a strongly irreversible system S_2 is emulated by a system S_1 which is not strongly irreversible, and we then use the definition of strongly irreversible system, and the definition of emulation, to deduce a contradiction. Notice that this proof employs *both* conditions (2c) and (2d) of the definition of emulation. That is, merging orbits cannot be emulated by a system which lacks this type of orbit because (i) each step of the emulated system is in fact emulated by the emulating system and (ii) each step of the emulating system which transforms states which correspond to the states of the emulated system emulates a step of this system.

Theorem 6

If S_2 is strongly irreversible and S_1 is not strongly irreversible, S_1 does not emulate S_2

proof:

suppose S_2 is strongly irreversible, and S_1 is not strongly irreversible; suppose for reductio that S_1 emulates S_2 . Since S_2 is strongly irreversible, there are x and y , $x \neq y$, such that, for some t and z , $g^t(x) = g^z(y)$ and, for all w , $g^w(x) \neq y$ and $g^w(y) \neq x$. Since S_1 emulates S_2 , from condition (2c) of definition 4, $u(g^t(x)) = h^{v(x-z)}(u(x))$ and $u(g^z(y)) = h^{v(y-z)}(u(y))$. Hence, $h^{v(x-z)}(u(x)) = h^{v(y-z)}(u(y))$. Since S_1 is not strongly irreversible, S_1 has no merging orbits. Therefore, either $\text{orb}(u(x)) \subseteq \text{orb}(u(y))$ or $\text{orb}(u(y)) \subseteq \text{orb}(u(x))$. Thus, there is a time $s \geq 0$ such that $h^s(u(x)) = u(y)$ or $h^s(u(y)) = u(x)$. Then, since S_1 emulates S_2 , from condition (2d) of definition 4, there

is a time w such that $g^w(x) = y$ or $g^w(y) = x$. But, for all w , $g^w(x) \neq y$ and $g^w(y) \neq x$. We have thus reached a contradiction, whence S_1 does not emulate S_2 q.e.d.

An immediate, and important, consequence of theorem 6 is that *all universal computational systems are strongly irreversible*. C is a universal computational system iff: C is a computational system which emulates all computational systems. Such a class is not empty, for there is a Turing machine which is universal in this sense²⁸. Furthermore, all universal computational systems must be strongly irreversible. This follows from their universality, from theorem 6, and from the fact that some computational systems are strongly irreversible. This result is also interesting because it highlights a necessary, *qualitative*, feature of the phase-portrait of all universal systems: all these systems must have *merging orbits*. More in general, the definition of universal computational system, theorem 6, and definition 4 imply that *a universal computational system has orbits of all possible types: periodic, eventually periodic, aperiodic, and merging*²⁹. If some type is missing, the system cannot be universal.

²⁸ If S is an arbitrary computational system, the transition function of a system S_1 isomorphic to S is computable by a Turing machine. We can thus modify this Turing machine to obtain a Turing machine which emulates S_1 . Also, there is a Turing machine which emulates any other Turing machine. From this, and the transitivity of the emulation relation, it follows that this machine emulates all computational systems.

²⁹ There are two types of merging orbits: merging and eventually periodic or merging and aperiodic. Both types must be present in a universal system. The proof that a system S_1 which lacks a type of orbit cannot emulate a system S_2 with that type of orbit is analogous to the proof of theorem 6. We assume first that S_1 emulates S_2 , and we then deduce a contradiction from the definition of that orbit type, and from the definition of emulation. These theorems thus show that the emulation relation preserves the qualitative features of the phase portrait of the emulated system. We will see shortly that a natural generalization of the emulation relation, the *realization relation*, does not satisfy this property.

Theorem 6 implies that no strongly irreversible system can be emulated by a reversible one. Nevertheless, some strongly irreversible systems can be implemented, or *realized*, by reversible systems. For example, Norman Margolus (1984) has proved that a certain reversible system is computationally universal. But I have just shown that all computationally universal systems are strongly irreversible, so that these two results seem to be inconsistent. This apparent paradox in fact shows that the emulation relation cannot be the relation which holds between an arbitrary computational system and a universal *reversible* system. I am now going to analyze this second relation, and I will always use the word "realization" for referring to it. The word "emulation", instead, always refers to the relation defined in section 4. Finally, I leave the terms "implementation" and "simulation" unanalyzed.

To get an intuitive understanding of the concept I want to analyze, think of the relation between a Turing machine and its physical realizations as usually described. We have seen in section 3 (examples 2.1 and 3.1) that an arbitrary Turing machine can be identified with any member of a class of isomorphic dynamical systems. None of these systems, however, can be a physical system, for two basic reasons. In the first place, all these systems evolve in discrete time steps, while physical systems evolve continuously. In the second place, the phase space of a Turing machine has a denumerable number of states, while the phase space of a physical system has the power of the continuum. Therefore, Turing machines are not physical systems. Nevertheless, they can be implemented, or

realized, by physical systems³⁰.

A standard realization of a Turing machine is a physical system which satisfies a certain macroscopic definition. The details of this definition may vary, but the story typically goes like this. First, the

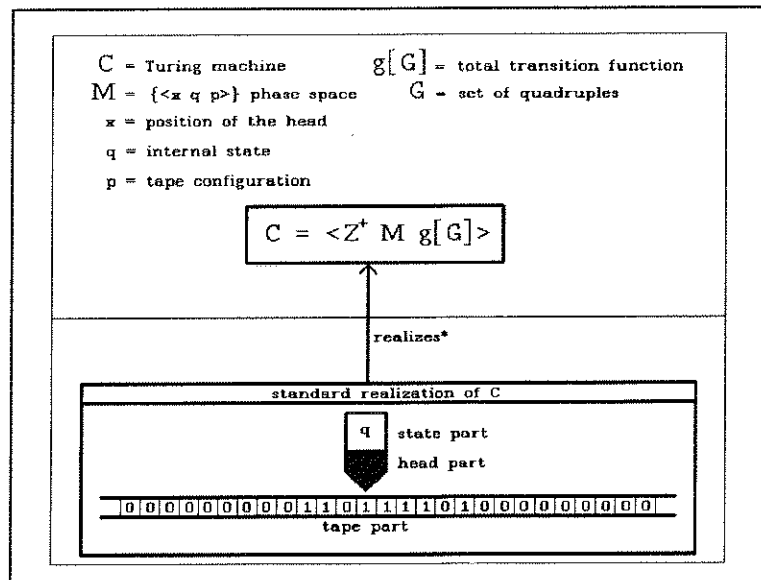


Figure 14 Standard realization of a Turing machine

system is deterministic, and it has *three parts*, which correspond to the *tape*, to the *head*, and to the *control unit* of the Turing machine. The *tape part* is a linear array of distinct addresses, and each address is in one of a finite number of different states, which correspond to the symbols of the Turing machine. The *head part* is a mechanism which is always located on exactly one tape address, has the capacity of changing the state of that address, and of moving to the address immediately to the left or to the right. Finally, the *state part* (or *control unit*) is in one of a finite number of mutually exclusive configurations, which correspond to the internal states of the Turing machine. Second, whenever the head part is set

³⁰ The same argument also applies to any computational system. No computational system is thus identical to a physical system. However, some physical systems realize some computational systems.

(or reset) on address \mathbf{x} , the state part is set (or reset) to configuration \mathbf{q} , and the tape part is set (or reset) to configuration \mathbf{p} , the head part will move to address \mathbf{x}' , the configuration of the state part will change to \mathbf{q}' , and the tape configuration will change to \mathbf{p}' , at a later time which only depends on \mathbf{x} , \mathbf{q} , and \mathbf{p} . Furthermore, \mathbf{x}' , \mathbf{q}' , and \mathbf{p}' correspond to the position, internal state, and tape configuration in which the Turing machine goes in one step when it is started on the position, internal state, and tape configuration which correspond to \mathbf{x} , \mathbf{q} , and \mathbf{p} ³¹.

If a physical system satisfies the definition stated above, then this system is in a definite relation with the Turing machine. I will call this relation "realization*". In the first place, notice that the definition of standard

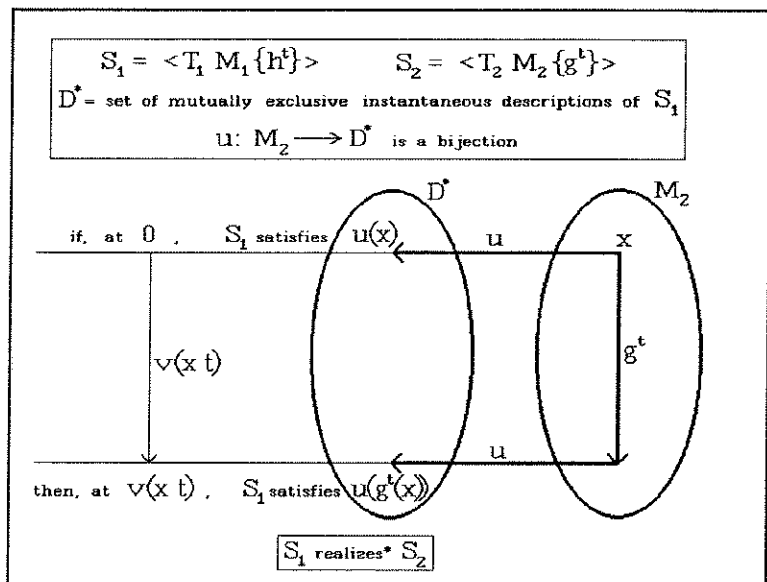


Figure 15 The realization relation (first version)

realization of a Turing machine involves a set of mutually exclusive instantaneous

³¹ I intend that, if t_0 is the time at which the physical system is set to \mathbf{x} , \mathbf{q} , and \mathbf{p} , and t_1 is the time at which the physical system resets itself to \mathbf{x}' , \mathbf{q}' , and \mathbf{p}' , then, for any time t between t_0 and t_1 , and any \mathbf{x} , \mathbf{q} , and \mathbf{p} , the physical system does not reset itself to \mathbf{x} , \mathbf{q} , and \mathbf{p} at t . This implies that, if x , q , and p are the position, internal state, and tape configuration of the Turing machine which correspond to \mathbf{x} , \mathbf{q} , and \mathbf{p} of the physical system, and $u(x, q, p)$ is the instantaneous description of the physical system "the head part is set (or reset) on address \mathbf{x} , the configuration of the state part is set (or reset) to \mathbf{q} , and the configuration of the tape part is set (or reset) to \mathbf{p} ", then there is a function $f(u(x, q, p)) =:$ the time necessary for resetting the head part, the state part, and the tape part to \mathbf{x}' , \mathbf{q}' , and \mathbf{p}' , where \mathbf{x}' , \mathbf{q}' , and \mathbf{p}' correspond to the position, internal state, and tape configuration in which the Turing machine goes in one step when it is started on x , q , and p . Furthermore, $f(u(x, q, p)) > 0$.

descriptions of the physical system. Each of these descriptions has the form: *the head part is set (or reset) on address x , the configuration of the state part is set (or reset) to q , and the configuration of the tape part is set (or reset) to p .* Obviously, there is a bijection u between the set of these descriptions D^* and the set of all total states of the Turing machine, so that each total state $\langle x q p \rangle$ can be identified with the corresponding instantaneous description $u(x q p)$ of the physical system. Furthermore, each transition of the Turing machine from total state $\langle x q p \rangle$ to total state $\langle x' q' p' \rangle$ is mirrored by a transition between the corresponding descriptions of the physical system. This implies that there is a function $v(\langle x q p \rangle t)$ which tells us how much time the physical system employs to mirror the step³² from $\langle x q p \rangle$ to $\langle x' q' p' \rangle = g[G]^t(x q p)$. Therefore, if the physical system satisfies description $u(x q p)$ at time 0 relative to initial state γ , then it must satisfy description $u(g[G]^t(x q p))$ at time $v(\langle x q p \rangle t)$ relative to initial state γ ³³. Also, since any description $u(x q p)$ of the physical system is identified with the total state $\langle x q p \rangle$ of the Turing machine, if the physical system satisfies description $u(x q p)$ at time 0 relative to initial state γ , and description $u(x' q' p')$ at time $t \geq 0$ relative to initial state γ , then there must be a step of the Turing machine

³² $G[g]$ is the transition function of the Turing machine, and $G[g]^t$ is an arbitrary t -advance (or state transition) of the Turing machine. $G[g]^t$ is obtained by iterating t times ($0 \leq t$) the transition function $G[g]$.

³³ We need to explicitly take into account the initial state γ of the physical system S because, depending on γ , S will in general satisfy different instantaneous descriptions at time t . This becomes obvious once we identify an instantaneous description $u(x q p)$ of S with a set of states of S , and we interpret "S satisfies description $u(x q p)$ at t relative to γ " as "there is $\alpha \in u(x q p)$ such that $h^t(\alpha) = \gamma$ ", where $\{h^t\}$ are the t -advances of S . See definition 6, and the discussion between definitions 5 and 6.

which corresponds to the transition from $u(x q p)$ to $u(x' q' p')$. That is, the following condition holds: if the physical system satisfies description $u(x q p)$ at time 0 relative to initial state γ , and it satisfies description $u(x' q' p')$ at time $t \geq 0$ relative to initial state γ , then there is a time w such that $g[G]^w(x q p) = \langle x' q' p' \rangle$ and $v(\langle x q p \rangle w) = t$. Finally, the function v assigns longer times to longer steps of the Turing machine, and it assigns time 0 to the step of length 0. All these properties of the relation which holds between a Turing machine and its standard realizations may also hold when we consider the relation between two *arbitrary* dynamical systems. Therefore, I abstractly define the realization* relation as follows:

Definition 5 (realization, first version)

S_1 realizes* S_2 iff:

- (1) $S_1 = \langle T_1 M_1 h(t \alpha) \rangle$ and $S_2 = \langle T_2 M_2 g(t x) \rangle$;
- (2) there are u and v such that:
 - (a) $u: M_2 \rightarrow D^*$, u is a bijection, and D^* is a set of mutually exclusive instantaneous descriptions of S_1 ;
 - (b) let $T_{1+} = \{t: t \in T_1 \text{ and } t \geq 0\}$; $v: M_2 \times T_{2+} \rightarrow T_{1+}$ such that:
 $v(x 0) = 0$; if $w < t$, then $v(x w) < v(x t)$;
 - (c) for any $t \in T_{2+}$, $\gamma \in M_1$, if S_1 satisfies $u(x)$ at time 0 relative to γ , then S_1 satisfies $u(g^t(x))$ at time $v(x t)$ relative to γ ;
 - (d) for any $t \in T_{1+}$, $\gamma \in M_1$, if S_1 satisfies $u(x)$ at time 0 relative to γ , and S_1 satisfies $u(y)$ at time t relative to γ , then there is a time w such that $g^w(x) = y$ and $v(x w) = t$.

I now verify that any standard realization of a Turing machine does satisfy definition 5.

Example 5.1 (any standard realization of a Turing machine realizes* that machine)

Let C be a Turing machine, and let $\rho(C)$ be a standard realization of C . Then, $\rho(C)$ realizes* C . By the definition of a standard realization of a Turing machine, $\rho(C)$ is a deterministic system; let $\rho(C) = S_1 = \langle T M \{h^i\} \rangle$. I identify C with $S_2 = \langle Z^+ W_2 g[G]_2 \rangle$ of example 2.1; condition (1) of definition 5 thus holds. Let $\langle x q p \rangle \in W_2$ be an arbitrary total state of C ; $u(x q p)$ is the following instantaneous description of $\rho(C)$: the head part of $\rho(C)$ is set (or reset) on address \mathbf{x} , the configuration of the state part of $\rho(C)$ is set (or reset) to \mathbf{q} , and the configuration of the tape part of $\rho(C)$ is set (or reset) to \mathbf{p} , where \mathbf{x} , \mathbf{q} and \mathbf{p} correspond to x , q , and p . Let $D^* = \{u(x q p) : \langle x q p \rangle \in W_2\}$; D^* is thus a set of mutually exclusive instantaneous descriptions of $\rho(C)$, and u is a bijection from W_2 to D^* . Condition (2a) of definition 5 is thus satisfied. Let $f: D^* \rightarrow T$ be defined by $f(u(x q p)) =:$ the time necessary for resetting the head-part, the state part, and the tape part of $\rho(C)$ to \mathbf{x}' , \mathbf{q}' , and \mathbf{p}' , where $g[G]_2(x q p) = \langle x' q' p' \rangle$ (see footnote 31). By the definition of $\rho(C)$, $f(u(x q p)) > 0$. Let $v(\langle x q p \rangle) = 0$ and, for all $k \in Z^+$, $v(\langle x q p \rangle k+1) = v(\langle x q p \rangle k) + f(u(g[G]_2^k(x q p)))$; condition (2b) of definition 5 thus holds, and condition (2c) is also obviously satisfied by u and v . To see that condition (2d) also holds, suppose $0 < t$, S_1 satisfies $u(x q p)$ at time 0 relative to γ , and S_1 satisfies $u(g[G]_2(x q p))$ at time t relative to γ . By the definitions of $\rho(C)$ and of D^* , for any instantaneous description $u(x q p) \in D^*$, and any $t: 0 < t_1 < t$, S does not satisfy $u(x q p)$ at time t relative to γ (see footnote 31). From this, and the definitions of u and v , condition (2d) of definition 5 follows. Therefore, $\rho(C)$ realizes* C .

If realize* successfully analyzes the relation between a dynamical system and its realizations, whenever a high level description of a physical realization of some dynamical system is given, realize* should hold between this dynamical system and any physical system which satisfies this description. Example 5.1 shows that this is in fact the case for the standard definition of a physical realization of a Turing machine. Many other examples can be found in computation theory. For instance, one can define a physical realization of a register machine (or of a cellular automaton) in the obvious way, and then prove that any system which satisfies this definition is in the relation realize* with the register machine (or with the cellular automaton). In general, realize* should also hold whenever a higher

level dynamical system is *reduced* to, but not identified with, a lower level one.

If a system S_1 realizes* a second system S_2 , each state of S_2 is univocally represented by an instantaneous description of S_1 , but definition 5 does not explicitly relate these instantaneous

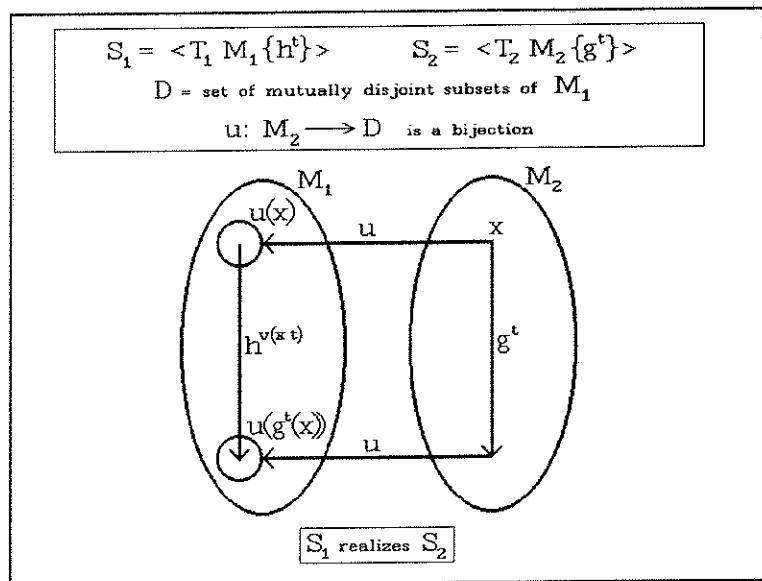


Figure 16 The realization relation (second version)

descriptions to the states and the t-advances of S_1 . However, each *state* of a dynamical system *completely describes the system at some instant*, so that any *set of states* can be thought as being expressed by *an instantaneous description of that system*. Conversely, any instantaneous description of a dynamical system always expresses a set of states of the system³⁴. Any *two mutually exclusive instantaneous descriptions* can thus be identified with *two disjoint subsets of the phase space*, and the phrase "system S_1 satisfies description $u(x)$ at time t relative to γ " simply means that "there is a state $\alpha \in u(x)$ such that $h^t(\gamma) = \alpha$ "³⁵, where $\{h^t\}$ are the t-advances (or state transitions) of S_1 . With this interpretation in mind we

³⁴ A *complete* instantaneous description expresses a set which only contains one state. An *incomplete* instantaneous description expresses a set which contains at least two states.

³⁵ Notice that, if t and γ are fixed, the state α is unique.

can thus put definition 5 into an equivalent form:

Definition 6 (realization, second version)

S_1 realizes S_2 iff:

- (1) $S_1 = \langle T_1 M_1 h(t \alpha) \rangle$ and $S_2 = \langle T_2 M_2 g(t x) \rangle$;
- (2) there are u and v such that:
 - (a) $u: M_2 \rightarrow D$, u is injective and surjective, and D is a set of mutually disjoint subsets of M_1 ;
 - (b) let $T_{1+} = \{t: t \in T_1 \text{ and } t \geq 0\}$; $v: M_2 \times T_{2+} \rightarrow T_{1+}$ such that:
 $v(x 0) = 0$; if $w < t$, then $v(x w) < v(x t)$;
 - (c) for any $t \in T_{2+}$, for any $\alpha \in u(x)$, there is $\beta \in u(g^t(x))$, such that $h^{v(x t)}(\alpha) = \beta$;
 - (d) for any $t \in T_{1+}$, for any $\alpha \in u(x)$, for any $\beta \in u(y)$, if $h^t(\alpha) = \beta$, then there is a time w such that $g^w(x) = y$ and $v(x w) = t$.

Definitions 6 and 5 are equivalent if an arbitrary set of states $u(x)$ in D is identified with the instantaneous description $u(x)$ in D^* , and the phrase " S_1 satisfies description $u(x)$ at time t relative to γ " is interpreted as "there is a state $\alpha \in u(x)$ such that $h^t(\gamma) = \alpha$ ". The equivalence of the two definitions of the realization relation is thus expressed by the following theorem:

Theorem 7

Definition 6 and definition 5 are equivalent under the interpretation:

- (1) $D^* = D$
- (2) S_1 satisfies description $u(x)$ at t relative to γ iff there is a state $\alpha \in u(x)$ such that $h^t(\gamma) = \alpha$

proof:

the thesis follows from definitions 5, 6, and from the interpretation stated above. For the details see the appendix q.e.d.

Two further requirements that the realization relation should certainly satisfy

are reflexivity and transitivity. I prove in the appendix that these properties are in fact entailed by definition 6. The proof that reflexivity holds is immediate. In fact, the realization relation reduces to the emulation relation once each state x of a system is identified with the set $\{x\}$ which contains that state. The proof that transitivity holds is analogous to the corresponding proof for the emulation relation (see theorem 2). However, the functions u_3 and v_3 must be defined in a somewhat different way. Suppose $S_1 = \langle T_1, M_1, g_1(t, x) \rangle$, $S_2 = \langle T_2, M_2, g_2(t, y) \rangle$, $S_3 = \langle T_3, M_3, g_3(t, z) \rangle$, and that S_1 realizes S_2 and S_2 realizes S_3 . Furthermore, let $u_1: M_2 \rightarrow D_1$, $v_1: M_2 \times T_{2+} \rightarrow T_{1+}$, $u_2: M_3 \rightarrow D_2$, and $v_2: M_3 \times T_{3+} \rightarrow T_{2+}$ satisfy definition 6. We then must show that there are $u_3: M_3 \rightarrow D$ and $v_3: M_3 \times T_{3+} \rightarrow T_{1+}$ which also satisfy definition 6. Recall that, to prove transitivity for the emulation relation, we defined $u_3(z) =: u_1(u_2(z))$ and $v_3(z, t) =: v_1(u_2(z), v_2(z, t))$. The problem is that, now, $u_2(z)$ is not a *state* of S_2 but, rather, a *set of states*, so that the two compositions $u_1(u_2(z))$ and $v_1(u_2(z), v_2(z, t))$ do not make sense. However, we can solve this problem with a simple trick. Instead of $u_2: M_3 \rightarrow D_2$, we use a function³⁶ $f: M_3 \rightarrow M_2$ which satisfies $f(z) \in u_2(z)$ and $f(g_3^t(z)) = g_2^{v_2(t, z)}(f(z))$, and we then define $u_3(z) =: u_1(f(z))$ and $v_3(z, t) =: v_1(f, v_2(z, t))$. It is then easy to prove that u_3 and v_3 satisfy definition 6. I give the details in the appendix.

Theorem 8

The realization relation is a quasi-ordering on the set of all dynamical systems

³⁶ The existence of such a function is guaranteed by the fact that u_2 and v_2 satisfy definition 6.

proof:
see the appendix

Finally, transitivity with respect to emulation, and transitivity with respect to isomorphism, immediately follow from the previous theorem:

Corollary 8.1

- (a) If S_1 realizes S_2 and S_2 emulates S_3 , then S_1 realizes S_3
- (b) if S_1 realizes S_2 and S_2 is isomorphic to S_3 , then S_1 realizes S_3

proof of (a):

notice that, if S_2 emulates S_3 , then S_2 realizes S_3 . From this and theorem 8 the thesis follows //q.e.d//

proof of (b):

notice that, if S_2 is isomorphic to S_3 , then S_2 emulates S_3 . From this and (a) the thesis follows //q.e.d// q.e.d.

Example 5.1 (continued)

In example 5.1 I have shown that, once an arbitrary Turing machine C is identified with the dynamical system $S_2 = \langle Z^+ W_2 g[G]_2 \rangle$, $\rho(C)$ realizes* C , where $\rho(C)$ is a standard realization of C . Corollary 8.1(b) and theorem 7 imply that this proof does not depend on the particular system I have chosen to represent C .

7. Virtual systems, the realizability of irreversible systems, and the existence of universal reversible systems

In computation theory it is customary to talk of the virtual computer implemented by a system. The implementing system may either be a physical computer, and the virtual

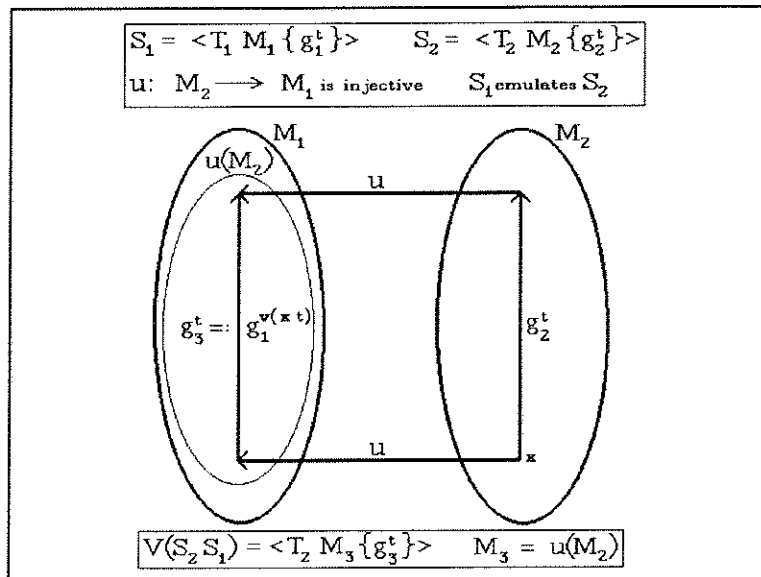


Figure 17 The virtual system $V(S_2, S_1)$ when S_1 emulates S_2

computer is supposedly defined in terms of the lower level workings of the implementing system. The relations of emulation and realization allow us to make this idea precise. *Whenever a system S_1 either emulates or realizes a second system S_2 , the states and the t -advances of S_1 , together with the functions u and v , can be used to define a third system $V(S_2, S_1)$ which turns out to be isomorphic to S_2 .* I will thus call this system the virtual S_2 in S_1 . The basic idea of the definition is that we may identify each state of S_2 with $u(y)$, where $u(y)$ is the state (or set of states) of S_1 which represents the state y of S_2 . We can then use $v(y, t)$ to define the t -advance g_3^t of the virtual system which corresponds to $g_2^t(y)$, where g_2^t is an arbitrary t -advance of S_2 . If $u(y)$ is a state of S_1 , that is, if S_1 emulates S_2 , $g_3^t(u(y))$ is simply defined as $g_1^{v(y, t)}(u(y))$.

If $u(y)$ is a set of states of S_1 , that is, if S_1 realizes S_2 , let us first choose an

arbitrary $x \in u(y)$, and then define $g_3^t(u(y))$ as the set in $\text{Im}(u)$ which contains $g_1^{v(y-t)}(x)$. Definition 6 ensures the existence and uniqueness of this set, and that g_3^t does not depend on the particular

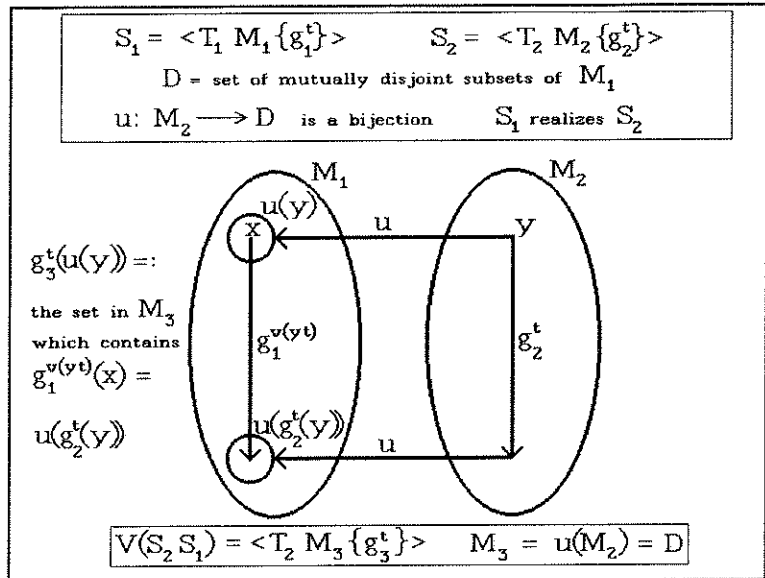


Figure 18 The virtual system $V(S_2 S_1)$ when S_1 realizes S_2

x we choose to represent $u(y)$. Below is the formal definition.

Definition 7 (virtual system)

$V(S_2 S_1)$ is the virtual S_2 in S_1 iff:

$S_1 = \langle T_1 M_1 g_1(t x) \rangle$, $S_2 = \langle T_2 M_2 g_2(t y) \rangle$, and $V(S_2 S_1) = \langle T_2 M_3 g_3(t z) \rangle$ satisfy:

- (1) either S_1 emulates S_2 or S_1 realizes S_2 and, if $T_2 = R, Q,$ or Z , then $T_1 = R, Q,$ or Z ;
- (2) for some u and v which either satisfy definition 4 or satisfy definition 6:
 - (a) $M_3 = \text{Im}(u)$;
 - (b) for all $y \in M_2$, if u and v satisfy definition 4, let $\chi(u(y)) = \sigma(u(y)) = u(y)$; if u and v satisfy definition 6, let $\chi(u(y)) \in u(y)$ and, for all $x: x \in u(y)$ for some y , let $\sigma(x) = u(y)$. Then, for all $t \in T_{2+}$, $g_3^t(u(y)) = \sigma(g_1^{v(y-t)}(\chi(u(y))))$. If $T_2 = R, Q,$ or Z , first define v also for negative instants: $v(x -t) =: -v(x t)$. Then set: $g_3^{-t}(u(y)) = \sigma(g_1^{v(y-t)}(\chi(u(y))))$.

I now prove that $V(S_2 S_1)$ is indeed a deterministic dynamical system, and that it is isomorphic to S_2 . This theorem is a straightforward consequence of the definitions of virtual system, dynamical system, isomorphism, emulation, and realization. In particular, the proof that $g_3^{w+t}(u(y)) = g_3^w(g_3^t(u(y)))$ employs the

equality $g_1^{v(y^{w+t})}(\chi(u(y))) = g_1^{v(g_2^{t(y)} w) + v(y^t)}(\chi(u(y)))$, which follows from the definition of $\chi(u(y))$ and from condition (2c) of the definitions of emulation (definition 4) and realization (definition 6).

Theorem 9

- (a) $V(S_2, S_1)$ is a deterministic dynamical system
- (b) $V(S_2, S_1)$ is isomorphic to S_2

proof of (a):

$$\begin{aligned}
 g_3^0(u(y)) &= \sigma(g_1^{v(y^0)}(\chi(u(y)))) = \sigma(g_1^0(\chi(u(y)))) = \sigma(\chi(u(y))) = u(y); \\
 g_3^{w+t}(u(y)) &= \sigma(g_1^{v(y^{w+t})}(\chi(u(y)))) = \\
 \sigma(g_1^{v(g_2^{t(y)} w) + v(y^t)}(\chi(u(y)))) &= \sigma(g_1^{v(g_2^{t(y)} w)}(g_1^{v(y^t)}(\chi(u(y)))) = \sigma(g_1^{v(g_2^{t(y)} w)}(\chi(\sigma(g_1^{v(y^t)}(\chi(u(y))))))) = \\
 g_3^w(\sigma(g_1^{v(y^t)}(\chi(u(y)))))) &= g_3^w(g_3^t(u(y))) \quad //q.e.d.//
 \end{aligned}$$

proof of (b):

first notice that $u: M_2 \rightarrow M_3$ is injective and surjective; furthermore, from (2b) of definition 7:

$$g_3^t(u(y)) = \sigma(g_1^{v(y^t)}(\chi(u(y)))) = u(g_2^t(y)) \quad //q.e.d.// \quad q.e.d.$$

We have seen above (theorem 6) that no strongly irreversible system can be emulated by a reversible one. Strongly irreversible systems, however, can be realized by reversible ones. I now prove an interesting general result: given an arbitrary irreversible system S , it is always possible to construct a reversible system $R(S)$ which realizes S . The basic idea of such construction is this. Suppose S is an irreversible system and that the state of S at the present time t is y . If the initial state was x , the state y has been achieved by going through a series of previous states (the actual history of x up to t) but, since S is irreversible, y could have been achieved through a history which starts from a different initial

state z . Thus, if we want to recover the past of y , we must take into account the actual history which produced y , not only the present state y . This observation leads to the idea of considering, for any irreversible system S , a corresponding reversible system $R(S)$ whose states are all the histories of an arbitrary initial state x up to an arbitrary time t . I will then prove that this system realizes S .

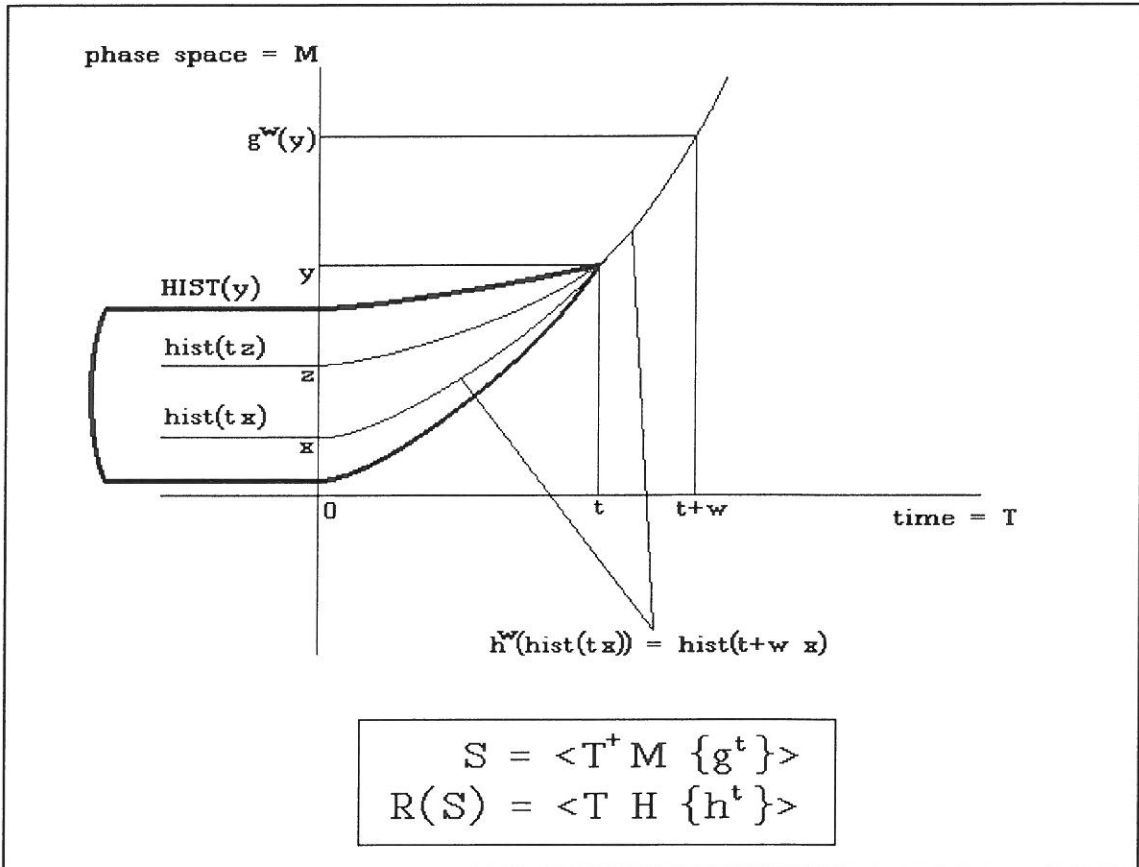


Figure 19 Construction of a reversible system $R(S)$ which realizes an irreversible system S

Let us first see the details of the construction of $R(S)$. Let $S = \langle T^+ M \{g^t\} \rangle$ be irreversible. Then, $R(S) = \langle T H \{h^t\} \rangle$ satisfies the following conditions. If $T^+ = R^+$, Q^+ , or Z^+ , then respectively $T = R$, Q , or Z . Let us define, for each $t \in T$ and $x \in M$, the history of x up to time t . This is in fact the state evolution which starts

with x up to the time t . Furthermore, let us add the pair $\langle w, x \rangle$ for each time $w < 0$. Formally:

$$\text{hist}(t, x) =: \{ \langle w, y \rangle : w \in T \text{ and } w \leq t \text{ and } y \in M \text{ and,} \\ \text{if } w \leq 0, y = x; \\ \text{if } w > 0, y = g^w(x) \}.$$

Let $H =: \{y : y = \text{hist}(t, x), \text{ for some } t \in T \text{ and } x \in M\}$. The group $\{h^t\}$ is then defined as follows:

$$h^w(\text{hist}(t, x)) =: \text{hist}(t+w, x), \text{ for all } x \in M \text{ and } w, t \in T.$$

Let us now verify that $R(S)$ is a reversible dynamical system:

$$h^0(\text{hist}(t, x)) = \text{hist}(t+0, x) = \text{hist}(t, x) \\ h^w(h^z(\text{hist}(t, x))) = h^w(\text{hist}(t+z, x)) = \text{hist}(t+z+w, x) = \text{hist}(t+w+z, x) = h^{w+z}(\text{hist}(t, x)).$$

I now prove that $R(S)$ realizes S , and that the virtual S in $R(S)$ is isomorphic to S . Obviously, the second part of the thesis follows from the first part and from theorem 9. The proof that $R(S)$ realizes S is based on the following idea. If y is an arbitrary state of S , consider all the histories which might have produced y at some time t . I indicate this set by "HIST(y)". Intuitively, since HIST(y) is the set of all histories which produce y at some t , we can identify y with HIST(y). To show that $R(S)$ realizes S we must exhibit two functions u and v which satisfy definition 6. Since we have identified y with HIST(y), the natural choices are $u(y) = \text{HIST}(y)$, and $v(y, t) = t$. It is then easy to prove that u and v satisfy definition 6. I give the details below.

Theorem 10 (any irreversible system can be realized by a reversible system)

For all irreversible systems S , there is a reversible system $R(S)$ such that $R(S)$ realizes S and $V(S R(S))$ is isomorphic to S

proof:

let $R(S)$ be the system defined above. For all $y \in M$, let $HIST(y) =: \{\text{hist}(t x) : \langle t y \rangle \in \text{hist}(t x), \text{ for some } t \in T \text{ and } x \in M\}$. I show first: if $y \neq z$, $HIST(y) \cap HIST(z) = \emptyset$. Suppose $y \neq z$, $\text{hist}(t x) \in HIST(y)$, and $\text{hist}(t x) \in HIST(z)$. Then, $\langle t y \rangle \in \text{hist}(t x)$ and $\langle t z \rangle \in \text{hist}(t x)$. By the definition of $\text{hist}(t x)$, $y = z$, against the hypothesis. Let us now define: $u(y) =: HIST(y)$; $v(y t) =: t$. I now show that u and v satisfy definition 6. Conditions (1), (2a), and (2b) obviously hold. As for condition (2c), suppose $\text{hist}(t x) \in u(y) = HIST(y)$; $h^{v(y w)}(\text{hist}(t x)) = h^w(\text{hist}(t x)) = \text{hist}(t+w x) \in HIST(g^w(y)) = u(g^w(y))$. I finally prove condition (2d). Suppose $\text{hist}(t x) \in u(a)$, $\text{hist}(w y) \in u(b)$, and $h^z(\text{hist}(t x)) = \text{hist}(w y)$. Hence, $\text{hist}(t+z x) = \text{hist}(w y)$. Therefore, $\text{hist}(t+z x) \in u(b)$. From this, the definition of u , and since $\text{hist}(t x) \in u(a)$, $g^z(a) = b$. Finally, by the definition of v , $v(a z) = z$. $R(S)$ thus realizes S . From this and theorem 9, $V(S R(S))$ is isomorphic to S q.e.d.

We have seen in section 6 that all universal computational systems are strongly irreversible. This result depends on the fact that we have defined universal systems by means of the emulation relation. If, instead, we use the realization relation, we obtain a more comprehensive concept of universal system. S is universal* iff: S realizes all computational systems. An immediate consequence of this definition, of theorem 10, and of corollary 8.1(a) is the *existence of universal* reversible* systems. In fact, if C is a universal Turing machine, C is irreversible³⁷ and C emulates all computational systems. Therefore, by theorem 10, $R(C)$ realizes C . Also, since the realization relation is transitive with respect to emulation (corollary 8.1(a)), $R(C)$ realizes all computational systems. It thus follows that $R(C)$ is a universal* reversible system.

³⁷ Recall that all universal computational systems are strongly irreversible. See the discussion after theorem 6.

8. Appendix

Theorem 1

The relation of isomorphism between dynamical systems is an equivalence relation

proof:

we must prove that the relation of isomorphism between dynamical systems is (a) reflexive; (b) symmetric; (c) transitive.

proof of (a):

let f be the identity function on the phase space; definition 2 is then obviously satisfied //q.e.d.//

proof of (b):

suppose S_1 is isomorphic to S_2 ; consider $f^1: M_2 \rightarrow M_1$; f^1 is injective and surjective, and for all $t \in T$ and $y \in M_2$:

$$g^t(f^1(y)) = g^t(f^1(f(x))) = g^t(x)$$

$$f^1(h^t(y)) = f^1(h^t(f(x))) = f^1(f(g^t(x))) = g^t(x)$$

therefore:

$$f^1(h^t(y)) = g^t(f^1(y)) \quad //q.e.d.//$$

proof of (c):

let $S_1 = \langle T M_1 g_1(t x) \rangle$, $S_2 = \langle T M_2 g_2(t y) \rangle$, $S_3 = \langle T M_3 g_3(t z) \rangle$; suppose S_1 is isomorphic to S_2 and S_2 is isomorphic to S_3 . Let $f_1: M_1 \rightarrow M_2$ and $f_2: M_2 \rightarrow M_3$ satisfy definition 2; consider $f_3 = f_2 \circ f_1$; f_3 is injective and surjective from M_1 to M_3 and, for all $t \in T$ and $x \in M_1$:

$$g_3^t(f_3(x)) = g_3^t(f_2(f_1(x))) = f_2(g_2^t(f_1(x))) = f_2(f_1(g_1^t(x))) = f_3(g_1^t(x)) \quad //q.e.d.// \quad q.e.d.$$

Theorem 2

The emulation relation is a quasi-ordering on the set of all deterministic dynamical systems

proof:

we must prove that the emulation relation is (a) reflexive, and (b) transitive

proof of (a):

let $u: M \rightarrow M$ be the identity function on M ; let $v: M \times T_+ \rightarrow T_+$ be defined by: $v(x t) = t$. Obviously, u and v satisfy conditions (2a) and (2b) of definition 4. Conditions (2c) and (2d) are also satisfied:

$$u(g^t(x)) = g^t(x) = g^{v(x t)}(u(x));$$

suppose $g^t(u(x)) = u(y)$; then $g^t(x) = y$, and $v(x t) = t$; //q.e.d.//

proof of (b):

let $S_1 = \langle T_1 M_1 g_1(t x) \rangle$, $S_2 = \langle T_2 M_2 g_2(t y) \rangle$, $S_3 = \langle T_3 M_3 g_3(t z) \rangle$.

Suppose S_1 emulates S_2 and S_2 emulates S_3 .

Let $u_1: M_2 \rightarrow M_1$, $u_2: M_3 \rightarrow M_2$, $v_1: M_2 \times T_{2+} \rightarrow T_{1+}$, $v_2: M_3 \times T_{3+} \rightarrow T_{2+}$ satisfy definition 2.

Let $u_3 =: u_1 \circ u_2$; u_3 is injective from M_3 to M_1 . Condition (2a) of definition 4 is thus satisfied.

Let $v_3: M_3 \times T_{3+} \rightarrow T_{1+}$ be defined by: $v_3(z t) =: v_1(u_2(z) v_2(z t))$.

v_3 satisfies conditions (2b) of definition 4:

$$v_3(z 0) = v_1(u_2(z) v_2(z 0)) = v_1(u_2(z) 0) = 0 \quad //q.e.d. \text{ first condition} //$$

suppose:

1. $t < w$

from the definition of v_3 :

$$2. v_3(z t) = v_1(u_2(z) v_2(z t))$$

from the definition of v_3 :

$$3. v_3(z w) = v_1(u_2(z) v_2(z w))$$

from 1 and since v_2 satisfies definition 4:

$$4. v_2(z t) < v_2(z w)$$

from 4 and since v_1 satisfies definition 4:

$$5. v_1(u_2(z) v_2(z t)) < v_1(u_2(z) v_2(z w))$$

from 5, 3 and 2:

$$6. v_3(z t) < v_3(z w) \quad //q.e.d. \text{ second condition} //$$

u_3 and v_3 satisfy condition (2c) of definition 4:

$$u_3(g_3^t(z)) = u_1(u_2(g_3^t(z))) =$$

$$u_1(g_2^{u_2(z)t}(u_2(z))) = g_1^{u_1(u_2(z) v_2(z t))}(u_1(u_2(z))) = g_1^{v_3(z t)}(u_3(z)) \quad //q.e.d. //$$

u_3 and v_3 satisfy condition (2d) of definition 4:

suppose:

$$1. g_1^t(u_3(x)) = u_3(y)$$

from 1 and the definition of u_3 :

$$2. g_1^t(u_1(u_2(x))) = u_1(u_2(y))$$

from 2 and since u_1 and v_1 satisfy definition 4:

$$3. \text{ there is } w: g_2^w(u_2(x)) = u_2(y) \text{ and } t = v_1(u_2(x) w)$$

from 3 and since u_2 and v_2 satisfy definition 4:

$$4. \text{ there is } s: g_3^s(x) = y \text{ and } w = v_2(x s)$$

from 4 and 3:

$$5. \text{ there is } s: g_3^s(x) = y \text{ and } t = v_1(u_2(x) v_2(x s))$$

from 5. and the definition of v_3 :

$$6. \text{ there is } s: g_3^s(x) = y \text{ and } t = v_3(x s) \quad //q.e.d. // \text{ Theorem 2 is thus proved } q.e.d.$$

Lemma 5.1

If S has eventually periodic orbits or S has merging orbits, there is t such that g^t is not injective

proof:

suppose:

1. S has eventually periodic orbits

let:

2. $\text{orb}(x)$ be eventually periodic

from 2 and the definition of eventually periodic orbit:

3. there are y and t : $g^t(x) = y$ and $\text{orb}(y)$ is periodic

from 3 and the definition of periodic orbit:

5. there is $w > 0$ such that $g^t(x) = y$ and $g^w(y) = y$

from 2 and the definition of eventually periodic orbit:

6. $\text{orb}(x)$ is not periodic

from 5:

7. $g^t(x) = y = g^w(y) = g^w(g^t(x)) = g^t(g^w(x))$

from 6, 5 and the definition of periodic orbit:

8. $g^w(x) \neq x$

from 8 and 7:

9. g^t is not injective //done with case 1//

suppose:

10. S has merging orbits

from 10, corollary 3.1, theorem 4, and the definition of quasi-reversible system:

11. there is t such that g^t is not injective //done with case 10// q.e.d.

Lemma 5.2

If there is t such that g^t is not injective, and S has no merging orbits, S has eventually periodic orbits

proof:

suppose:

1. there are t, w, x, y, z : $x \neq y$, $g^t(x) = g^t(y) = z$, and $g^w(x) = y$

We first show that $\text{orb}(z)$ is periodic

from 1:

2. $z = g^t(y) = g^t(g^w(x)) = g^w(g^t(x)) = g^w(z)$, whence $\text{orb}(z)$ is periodic //q.e.d.//

We now show that $\text{orb}(x)$ is not periodic

from 2 and 1:

3. $x \neq z$

suppose for reductio:

4. there is s : $g^s(x) = x$

suppose:

5. $s = t$

from 4, 5, 1 and 3:

$$6. x = g^s(x) = g^t(x) = z \text{ and } x \neq z$$

//done with case 5//

suppose:

$$7. s > t$$

from 4, 7, and 1:

$$8. x = g^s(x) = g^{s-t}(g^t(x)) = g^{s-t}(z)$$

from 2, 7, and 8:

$$9. z = g^w(z) = g^{t+w-s+s-t}(z) = g^{t+w-s}(g^{s-t}(z)) = g^{t+w-s}(x)$$

from 8, 9, 2 and 3:

$$10. x = g^{s-t}(g^{t+w-s}(x)) = g^{s-t+t+w-s}(x) = g^w(x) = z \text{ and } x \neq z$$

//done with case 7//

suppose:

$$11. s < t$$

from 4 and 11:

$$12. x = g^s(x) = g^{t-s}(g^s(x)) = g^{t-s}(x)$$

from 1, 11 and 12:

$$13. z = g^t(x) = g^s(g^{t-s}(x)) = g^s(x)$$

from 13, 12 and 3:

$$14. z = g^s(x) = x \text{ and } x \neq z$$

//done with case 11 done with 4 q.e.d.//

Lemma 5.2 is thus proved q.e.d.

Theorem 7

Definition 5 and definition 6 are equivalent under the interpretation:

(1) $D^* = D$;

(2) S_1 satisfies description $u(x)$ at time t relative to γ iff there is $\alpha \in u(x)$ such that $h^t(\gamma) = \alpha$

proof:

suppose:

1. $D^* = D$

2. S_1 satisfies description $u(x)$ at time t relative to γ iff there is $\alpha \in u(x)$ such that $h_t(\gamma) = \alpha$

suppose:

3. S_1 realizes* S_2

from 1, 3, and conditions (1), (2a) and (2b) of definition 5:

4. conditions (1), (2a), and (2b) of definition 6 are satisfied

from 2, 3, and condition (2c) of definition 5:

5. for any $t \in T_{2+}$, $\gamma \in M_1$, if there is $\alpha \in u(x)$ such that $h^0(\gamma) = \alpha$, then there is $\beta \in u(g^t(x))$ such that $h^{v(x,t)}(\gamma) = \beta$

from 5:

6. for any $t \in T_{2+}$, for any $\alpha \in u(x)$, there is $\beta \in u(g^t(x))$, such that $h^{v(x,t)}(\alpha) = \beta$; hence, condition (2c) of definition 6 is satisfied

from 2, 3, and condition (2d) of definition 5:

7. for any $t \in T_{1+}$, $\gamma \in M_1$, if there is $\alpha \in u(x)$ such that $h^0(\gamma) = \alpha$, and there is $\beta \in u(y)$ such that $h^t(\gamma) = \beta$, then there is a time w such that $g^w(x) = y$ and $v(x,w) = t$

from 7:

8. for any $t \in T_{1+}$, for any $\alpha \in u(x)$, if there is $\beta \in u(y)$ such that $h^t(\alpha) = \beta$, then there is a time w such that $g^w(x) = y$ and $v(x w) = t$;

from 8:

9. for any $t \in T_{1+}$, for any $\alpha \in u(x)$, for any $\beta \in u(y)$, if $h^t(\alpha) = \beta$, then there is a time w such that $g^w(x) = y$ and $v(x w) = t$; hence, condition (2d) of definition 6 is satisfied

from 4, 6, and 9:

10. S_1 realizes S_2

//done with hypothesis 3//

suppose:

11. S_1 realizes S_2

from 1, 11, and conditions (1), (2a) and (2b) of definition 6:

12. conditions (1), (2a), and (2b) of definition 5 are satisfied

from 2, 11, and condition (2c) of definition 6:

13. for any $t \in T_{2+}$, for any $\alpha \in u(x)$, S_1 satisfies $u(g^t(x))$ at $v(x t)$ relative to α
from 2:

14. condition (2c) of definition 5 is equivalent to 13

from 13 and 14:

15. condition (2c) of definition 5 is satisfied

from 11, and condition (2d) of definition 6:

16. for any $t \in T_{1+}$, for any $\alpha \in u(x)$, if there is $\beta \in u(y)$ such that $h^t(\alpha) = \beta$, then there is a time w such that $g^w(x) = y$ and $v(x w) = t$

from 16 and 2:

17. for any $t \in T_{1+}$, for any $\alpha \in u(x)$, if S_1 satisfies $u(y)$ at t relative to α , then there is a time w such that $g^w(x) = y$ and $v(x w) = t$

from 2:

18. condition (2d) of definition 5 is equivalent to 17

from 17 and 18:

19. condition (2d) of definition 5 is satisfied

from 12, 15, and 19:

20. S_1 realizes* S_2

//done with hypothesis 11// q.e.d.

Theorem 8

The realization relation is a quasi-ordering on the set of all dynamical systems

proof:

we must prove:

(a) S realizes S

(b) if S_1 realizes S_2 and S_2 realizes S_3 , then S_1 realizes S_3

proof of (a):

for any state x of S , let $u(x) =: \{x\}$. Notice that, by identifying x with $\{x\}$, the realization relation reduces to the emulation relation. Therefore, by theorem 2, the realization relation is reflexive //q.e.d.//

proof of (b):

let $S_1 = \langle T_1 M_1 g_1(t x) \rangle$, $S_2 = \langle T_2 M_2 g_2(t y) \rangle$, $S_3 = \langle T_3 M_3 g_3(t z) \rangle$.

Suppose S_1 realizes S_2 and S_2 realizes S_3 .

Let $u_1: M_2 \rightarrow D_1$ and $v_1: M_2 \times T_{2+} \rightarrow T_{1+}$ satisfy definition 6.

Let $u_2: M_3 \rightarrow D_2$ and $v_2: M_3 \times T_{3+} \rightarrow T_{2+}$ satisfy definition 6.

For all $z \in M_3$ and $t \in T_3$, let $f: M_3 \rightarrow M_2$ satisfy $f(z) \in u_2(z)$ and $f(g_3^t(z)) = g_2^{v_2(z,t)}(f(z))$.

For all $z \in M_3$, let $u_3(z) = u_1(f(z))$. Let D be the image of u_3 . By definition, D is a set of mutually disjoint subsets of M_1 , and u_3 is injective and surjective from M_3 to D . Condition (2a) of definition 6 is thus satisfied.

Let $v_3: M_3 \times T_{3+} \rightarrow T_{1+}$ be defined by: $v_3(z t) = v_1(f(z) v_2(z t))$.

v_3 satisfies conditions (2b) of definition 6:

$$v_3(z 0) = v_1(f(z) v_2(z 0)) = v_1(f(z) 0) = 0 \quad //q.e.d. \text{ first condition} //$$

suppose:

1. $t < w$

from the definition of v_3 :

2. $v_3(z t) = v_1(f(z) v_2(z t))$

from the definition of v_3 :

3. $v_3(z w) = v_1(f(z) v_2(z w))$

from 1 and since v_2 satisfies definition 6:

4. $v_2(z t) < v_2(z w)$

from 4 and since v_1 satisfies definition 6:

5. $v_1(f(z) v_2(z t)) < v_1(f(z) v_2(z w))$

from 5, 3 and 2:

6. $v_3(z t) < v_3(z w) \quad //q.e.d. \text{ second condition} //$

u_3 and v_3 satisfy condition (2c) of definition 6:

we must prove: for all $\alpha \in u_3(z)$, $g_1^{v_3(z,t)}(\alpha) \in u_3(g_3^t(z))$

from the definition of v_3 :

1. $v_3(z t) = v_1(f(z) v_2(z t))$

from the definitions of u_3 and f :

2. $u_3(g_3^t(z)) = u_1(f(g_3^t(z))) =$

$$u_1(g_2^{v_2(z,t)}(f(z)))$$

from the definition of u_3 :

3. $u_3(z) = u_1(f(z))$

since u_1 and v_1 satisfy condition (2c) of definition 6:

4. for all $\alpha \in u_1(f(z))$,

$$g_1^{v_1(f(z) v_2(z,t))}(\alpha) \in u_1(g_2^{v_2(z,t)}(f(z)))$$

from 1, 2, 3, and 4:

5. for all $\alpha \in u_3(z)$,

$$g_1^{v_3(z,t)}(\alpha) \in u_3(g_3^t(z)) \quad //q.e.d. //$$

u_3 and v_3 satisfy condition (2d) of definition 6:

suppose:

1. $\alpha \in u_3(x)$, $\beta \in u_3(y)$, and $g_1^v(\alpha) = \beta$

from 1 and the definition of u_3 :

2. $\alpha \in u_1(f(x))$, $\beta \in u_1(f(y))$, and $g_1^v(\alpha) = \beta$

from 2 and since u_1 and v_1 satisfy condition (2d) of definition 6:

3. there is w : $g_2^w(f(x)) = f(y)$ and $t = v_1(f(x) w)$

from 3, from the definition of f , and since u_2 and v_2 satisfy condition (2d) of definition 6:

4. there is s : $g_3^s(x) = y$ and $w = v_2(x s)$

from 4 and 3:

5. there is s : $g_3^s(x) = y$ and $t = v_1(f(x) v_2(x s))$

from 5 and the definition of v_3 :

6. there is s : $g_3^s(x) = y$ and $t = v_3(x s)$ //q.e.d.// Theorem 8 is thus proved q.e.d.

Chapter 2

GENERALIZED COMPUTATIONAL SYSTEMS

1. Introduction
 2. Pattern fields and generalized Turing machines
 3. The relation between generalized Turing machines and ordinary ones
 4. Is a non-recursive pattern field necessary for computing non-recursive functions?
 5. Generalized computational systems on regular pattern fields
-

1. Introduction

The definition of a computational system which I have adopted in the first chapter presupposes that the class of all effective transformations of finite symbol structures be reducible to the class of the functions computable by a Turing machine. This identification is known as *Turing's thesis*. This thesis is usually justified in two different ways. First, one gives an informal argument which shows that any *symbolic* computation which can be performed by using paper and pencil can always be reduced to a series of operations of a Turing machine. The best justification of this type is still the argument which Turing put forth in the thirties (Turing 1965, sec. 9). The justification of the second type is instead based on a series of theorems of computability theory. These theorems show that all known computational systems can only compute numeric functions which can also be computed by a Turing machine. These numeric functions are all, and only, the

(partial) recursive functions. This second argument thus provides a direct justification of *Church's thesis*, which affirms that all the effective *numeric* functions are (partial) recursive. Nevertheless, these theorems also provide an indirect confirmation of Turing's thesis, which is more general. In fact, any numeric function can always be thought as a transformation of finite symbol structures. Therefore, if Church's thesis turned out to be false, Turing's thesis should be rejected as well.

Turing presented his machines as an idealization of a human being which transforms symbols by means of a specified set of rules. Turing proposed four hypotheses: **(1)** the capacity of recognizing, transforming, and memorizing symbols and rules is finite. It thus follows that any transformation of a complex symbol must always be reduced to a series of simpler transformations. These operations on elementary symbols are of three types: recognizing a symbol, replacing a symbol and, finally, shifting the attention to a symbol which is contiguous to the symbol which has been considered earlier. **(2)** The series of elementary operations which are in fact executed is determined by three factors. First, the symbol which the subject considers at a given time; second, the subject's mental state when he considers that symbol; third, a rule chosen from a finite number of alternatives. Turing also assumed that the number of possible mental states is finite¹, and that the rules are of two different types: [a] if the symbol

¹ This follows from the hypothesis that the capacity of memorizing symbols is finite. In fact, if the number of mental states were infinite, each of them might be used to memorize a different symbol.

considered at a given time is a_i , and the mental state at that time is q_j , replace a_i by a_k , and change mental state from q_j to q_n ; [b] if the symbol considered at a given time is a_i , and the mental state at that time is q_j , shift the attention to a contiguous symbol (to the right, to the left, above, below, etc.), and change mental state from q_j to q_n . **(3)** The elementary symbols are written on a support with an infinite capacity. This support (for example a sheet of paper which can be extended in any direction as needed) is divided in cells, and each cell may contain, at most, one elementary symbol. The number of cells which actually contain symbols is finite. **(4)** All different types of support can always be reduced to a tape divided in an infinite number of cells, and this reduction does not limit the computational capacity.

If we think of a transformation of symbols which a human being can perform by applying a finite number of rules, the first three of Turing's hypotheses are natural idealizations. The adequacy of the fourth hypothesis, however, is not so immediate. In fact, we can imagine of having a support whose division in cells is very complicated, and which also satisfies the following condition. Each cell communicates with a fixed number of other cells or, in other words, for an arbitrary cell, there are exactly m different *types* of path which lead to other cells. Now, if each type of path can always be distinguished from the others, we could use this support to perform a symbolic computation which satisfies Turing's first three hypotheses. The rules would be of the two forms: [a] if you are considering cell x whose symbol is a_i , and your mental state is q_j , replace a_i with a_k and change

state from q_i to q_n ; [b] if you are considering cell x whose symbol is a_i , and your mental state is q_j , consider next the cell at the end of path number r ($1 \leq r \leq m$), and change state from q_j to q_n .

More concretely, we can think of a magic castle with an infinite number of rooms, each room with the same number of doors. Suppose the doors of each room are numbered, and that each door leads to a corridor which ends in some other room. There may be magic doors: if you are in a room, and follow the corridors which correspond to these doors, you will reach the very same room. Also, two different corridors may end in the same room. Imagine that, on the floor of a finite number of rooms, a strange symbol is designed. You enter the castle. Unfortunately, you will be allowed to go out only after you have completed a very tedious job. You are given a finite list of instructions, and a spell modifies your mind so that it can only assume a finite number of different states. Your job consists in looking at the symbol on the floor and noticing your mental state, then comparing the mental state and the symbol with the list, finding the corresponding instruction, and executing it; if you cannot find any matching instruction, your job is finished. You only have two types of instructions: if your mental state is q_i and the symbol on the floor is a_i , change the symbol to a_k and your state to q_n ; if your state is q_j and the symbol is a_i , go through door number r and change your state to q_n . No two different instructions begin with the same pair state-symbol. It is clear that, even if the design of the castle is extremely complex, you will be able to perform your job. There is only one hitch: your task might never end. But this

sometimes happens when you enter magic castles ...

Now, this is the interesting question: if the topography of the castle is sufficiently complex, could it happen that the symbolic transformations performed in it are not reducible to those of a simpler topography? In particular, are we sure that *any* topography is always reducible to the extremely simple structure of a linear arrangement of cells? The standard answer to this question is based on empirical considerations. It can be proved that many different types of topography are in fact equivalent. For example, we can let a Turing machine operate on a checkerboard or, more in general, on a n -dimensional regular lattice. We can then show that a Turing machine of this type is not more powerful than an ordinary one. These proofs are then usually taken as conclusive evidence that, *in any case*, the computational power of Turing machines cannot change. Nevertheless, this argument is not convincing. The topographies for which we can prove the invariance are in fact quite simple. But what happens if the cells are connected in arbitrarily complex ways?

The first goal of this chapter is to give a definite answer to this question. One of the surprising results is that some Turing machines which operate on sufficiently complex topographies *are capable of computing numeric functions which are not recursive*. It thus follows that these machines are indeed more powerful than ordinary Turing machines, so that Turing's *fourth assumption* is false.

The basic ideas which permit to prove these facts are extremely simple. In the first place, given an arbitrary topography on which a Turing machine operates, we

need an effective way of representing numbers as some particular configurations (or *patterns*) of symbols. For example, if the topography is just a linear tape, number n is usually represented by a block of $n+1$ *consecutive* 1s. We thus need a similar convention for the general case. Notice now that the usual convention for ordinary Turing machines is based on the fact that the squares can be effectively enumerated by means of the relations *Left* and *Right*². We can then effectively define a block of $n+1$ *consecutive* 1s by means of this enumeration³. Analogously, if there were an *effective* way of enumerating the cells of an arbitrary topography *by means of its relational structure*, we could then use this enumeration to effectively specify the form of a pattern which represents number n . In fact, we could just stipulate, if we wished, that number n is represented by $n+1$ 1s, where the first 1 is on the first cell, the second 1 is on the second cell, ... the $n+1$ st 1 is on the $n+1$ st cell, and all other cells are blank.

Let us call any enumeration of the cells which depends effectively on the relational structure of a topography an *intrinsic enumeration*. The second observation consists in realizing that there is a large class of topographies for which an intrinsic enumeration of the cells exists. Suppose a Turing machine is started on a topography whose cells are all blank, and that this machine will then

² For example, we can effectively enumerate the squares by means of the relations *Left* and *Right* as follows: $e(0)$ =: an arbitrary square x ; $e(1)$ =: the square to the left of x ; $e(n)$ =: the square to the right of square $e(n-2)$, if n is even; $e(n)$ =: the square to the left of square $e(n-2)$, if n is odd and $n > 1$.

³ For example, let e be the enumeration defined in footnote 2. Two squares x and y are consecutive just in case the numbers n and m such that $e(n) = x$ and $e(m) = y$ satisfy one of the following conditions: (a) both n and m are even, and $n = m+2$ or $m = n+2$; (b) both n and m are odd, and $n = m+2$ or $m = n+2$; (c) $n = 0$, and $m = 1$ or $m = 2$; (d) $m = 0$, and $n = 1$ or $n = 2$.

move from cell to cell in such a way that *it will visit all the cells*. Obviously, this computation of the Turing machine generates an enumeration of the cells, so that it makes sense to talk of the first, the second, the third ... cell reached by the machine. Furthermore, this enumeration is intrinsic, for it depends on the relational structure of the topography in an effective way. Therefore, any topography for which such a Turing machine exists allows us to effectively represent numbers as patterns⁴. I call any such a topography a *regular pattern field*.

The third observation consists in realizing that there are some regular pattern fields which contain a non-recursive power hidden in the connections between their cells. In fact, I have assumed that each cell is connected to other cells by m different paths. Each of these paths can thus be identified with a function f_i ($1 \leq i < m$) whose argument is a cell, and whose value is another cell. If e is the intrinsic enumeration of the pattern field generated by a fixed Turing machine, each cell corresponds to a number, so that the function f_i corresponds to a numeric function f_{ei} . The point now is that *the numeric function f_{ei} may not be recursive*. If this is the case, the cells are connected in an essentially non-recursive way, and it turns out that this non-recursive power may then be used for computing numeric functions which are not recursive.

The next three sections of this chapter make the previous observations precise, and prove five general theorems about the computational power of Turing

⁴ For example, we can stipulate that number n is represented by $n+1$ 1s, where the first 1 is on the first cell, the second 1 is on the second cell, ..., the $n+1$ st 1 is on the $n+1$ st cell reached by the Turing machine, and all other cells are blank.

machines which operate on arbitrary topographies. The detailed plan of these sections is the following. In the second section, I define the concept of a *pattern field* and of a *generalized Turing machine* which operates on it. Formally, a pattern field is a pair $F = \langle U \{f_i\} \rangle$ where U is a denumerable set, and each f_i ($1 \leq i < n$) is a function from U to U . A Turing machine which operates on F is completely analogous to an ordinary Turing machine, except that it can move in n different ways, which correspond to the functions f_i . Therefore, a quadruple which specifies to move from the current cell to another one has the form $q_r a_i f_i q_s$. I then define *regular pattern fields*⁵, and I show how numbers can be effectively represented as patterns of symbols when an intrinsic enumeration is fixed. This allows me to define the concept of a *numeric COMPUTABLE function* relative to a fixed intrinsic enumeration of the pattern field (definition 6). I then show that all the numeric functions computable by an ordinary Turing machine with the usual conventions satisfy this definition (example 6.1).

In the third section, I first introduce the distinction between *recursive and non-recursive* pattern fields. When an intrinsic enumeration e is fixed, a regular pattern field $F = \langle U \{f_i\} \rangle$ is recursive just in case all the numeric functions $\{f_{e_i}\}$ which correspond to the functions $\{f_i\}$ are recursive. I then prove four theorems. The first states that there is a recursive pattern field on which all the (partial)

⁵ $F = \langle U \{f_i\} \rangle$ is a regular pattern field just in case there is an intrinsic enumeration e of F . Any intrinsic enumeration e is generated by a generalized Turing machine C which, when started in some internal state q_r and some node $x \in U$ of a completely blank pattern, reaches all the nodes in U .

recursive functions are COMPUTABLE (theorem 1)⁶. The second affirms that any function which is not (partial) recursive is COMPUTABLE on some, appropriately chosen, regular pattern field. However, the pattern field which permits this computation turns out to be non-recursive (theorem 2). I then consider the relation between COMPUTABILITY on regular pattern fields and the usual concept of *relative computability* (see Davis 1958), and I show that all the (partial) recursive functions relative⁷ to $\{f_i\}$ are COMPUTABLE on⁸ $F = \langle Z^+ \{s v\} \cup \{f_i\} \rangle$ relative to the identity function on Z^+ (theorem 3).

When an intrinsic enumeration e of a regular pattern field $F = \langle U \{f_i\} \rangle$ is fixed, e determines a set of numeric functions $\{f_{e_i}\}$ which correspond to the neighbor functions $\{f_i\}$. It is thus natural to ask whether the numeric functions $\{f_{e_i}\}$ are COMPUTABLE on F relative to e . Theorem 4 shows that this is in fact true. An immediate consequence of this theorem is that *being a regular, but non-recursive, pattern field is a sufficient condition for COMPUTING non-recursive functions* (corollary 4.1). In fact, since the pattern field F is not recursive, at least one of the numeric functions $\{f_{e_i}\}$ must be non-recursive. By theorem 4, however, this function is COMPUTABLE on F relative to e .

⁶ This pattern field turns out to be the tape of an ordinary Turing machine.

⁷ Let $\{f_i\}$ be a finite set of numeric functions. A numeric function g is (partial) recursive relative to $\{f_i\}$ just in case g can be obtained from f_i and the set of the basic recursive functions (successor, zero function, and identity functions) by applying a finite number of times the operations of primitive recursion, composition, and minimization.

⁸ $F = \langle Z^+ \{s v\} \cup \{f_i\} \rangle$ is the regular pattern field such that Z^+ are the non-negative integers, s is the successor function, v is the predecessor function (where $v(0) =: 0$), and $\{f_i\}$ is an arbitrary finite set of numeric functions.

In the fourth section, I consider the converse problem: *is a regular, but non-recursive pattern field, necessary for COMPUTING non recursive functions?* The answer to this question is provided by theorem 5. This theorem states that any numeric function COMPUTABLE on a regular pattern field $F = \langle U \{f_i\} \rangle$ relative to an arbitrary intrinsic enumeration e is (partial) recursive relative to the numeric functions $\{f_{e_i}\}$ which correspond to the neighbor functions $\{f_i\}$. In particular, if F is recursive, all the numeric functions $\{f_{e_i}\}$ are recursive, so that the class of the (partial) recursive functions relative to $\{f_{e_i}\}$ is identical to the class of the (partial) recursive functions. It thus follows that, for any recursive pattern field F , and any intrinsic enumeration e , any numeric function COMPUTABLE on F relative to e is (partial) recursive. Therefore, *being a non-recursive pattern field is a necessary condition for COMPUTING non-recursive functions* (corollary 5.3). A second consequence of theorem 5 (and of theorem 3) is that *the class of the (partial) recursive functions relative to $\{f_i\}$ is identical the class of the functions COMPUTABLE on⁹ $\langle Z^+ \{s v\} \cup \{f_i\} \rangle$ relative to the identity function on Z^+* (corollary 5.4).

Finally, in the fifth section, I consider the implications of these results for the definition of a computational system which I have given in chapter 1 (def. 3). That definition presupposes that the class of all effective transformations of finite symbol structures be reducible to the class of the functions computable by an ordinary

⁹ $F = \langle Z^+ \{s v\} \cup \{f_i\} \rangle$ is the regular pattern field such that Z^+ are the non-negative integers, s is the successor function, v is the predecessor function (where $v(0) =: 0$), and $\{f_i\}$ is an arbitrary finite set of numeric functions.

Turing machine. The results of sections 3 and 4, however, show that *the concept of an effective transformation of finite symbol structures is not absolute, but instead depends on the relational structure of the infinite support on which the elementary symbols are written*. In fact, if a generalized Turing machine operates on a regular but non-recursive pattern field, its transformations of finite symbol structures may permit the computation of non-recursive functions. On the other hand, Turing machines which operate on recursive pattern fields can only compute recursive functions. The goal of the fifth section is to give a more general definition of a computational system which takes into account these results. I thus define the concept of a *generalized computational system on a regular pattern field F* (definition 8). This definition is completely analogous to definition 3 of chapter 1, except that I now require that the transition function of an isomorphic system be COMPUTABLE on F relative to any intrinsic enumeration e , and that a characteristic function of its phase space be also COMPUTABLE on F relative to e . This new concept turns out to be a generalization of the previous one, in the sense that definition 8 reduces to definition 3 of chapter 1 when F is identified with the tape of an ordinary Turing machine. Finally, as a partial confirmation of the material adequacy of definition 8, I prove that all generalized Turing machines on any regular pattern field are generalized computational systems on that pattern field (theorem 6).

2. Pattern fields and generalized Turing machines

A finite symbol structure is a spatial arrangement of a certain number of elementary symbols. A typical example of a finite symbol structure is a string of symbols. In this case there are only two relations between the symbols: left and right. We can obtain richer structures if we add other relations like above and below. For instance, a word with superscripts and subscripts conforms to this model. I will call an arbitrary finite symbol structure a *pattern*¹⁰. John Winnie suggested thinking of the relational structure of a pattern as an infinite space, or a *pattern field*, in which the pattern is embedded. Patterns are usually taken to be *finite*, however, the field in which they occur is usually *infinite*. A typical example of pattern field is the infinite tape of an ordinary Turing machine. Another example is the infinite checkerboard of a bidimensional Turing machine¹¹. John Winnie pointed out that a pattern field basically consists of an infinite set of locations, or *nodes*, with a neighborhood structure which allows a finite machine to find its way from one location to its neighbors, by following a finite set of instructions and distinguishing, at most, a finite set of differences. Formally, a pattern field can be defined as follows:

¹⁰ The word "pattern" is used in a similar sense by Smith (1971). See in particular definition 3. According to Smith, a pattern manipulation system is a pair $\langle P, v \rangle$, where P is an effectively indexed (i.e. there is an effective enumeration of P) set of patterns, and v is a recursive function which maps pattern indexes to pattern indexes. Computational systems on regular pattern fields (see sec. 5, def. 8) are more general than the pattern manipulation systems defined by Smith. Smith's patterns turn out to be a special case of the patterns on regular pattern fields which I will define later.

¹¹ Turing machines which operate on an infinite checkerboard have been discussed in Dewdney (1989). The only difference between these machines and ordinary ones is that they can also move upwards and downwards.

Definition 1 (pattern field of dimension n , or the magic castle)

$F = \langle U \{f_i\} \rangle$ is a pattern field of dimension n iff:

- (1) U is a denumerable set; the elements of U are called the nodes of F ;
- (2) $\{f_i\}$ is a set of n ($n > 0$) functions from U to U ; f_i is called a neighbor function of F .

Example 1.1 (a tape infinite in one direction is a pattern field)

Let $U =$: the set of all squares of a tape infinite in one direction (say to the right), and let $\{L R\}$ satisfy: $L(x) =$: the square to the left of x , if x is not the leftmost square; $L(x) = x$, otherwise. $R(x) =$: the square to the right of x . Then, by definition 1, $F_1 = \langle U \{L R\} \rangle$ is a pattern field of dimension 2.

Example 1.2 (a tape infinite in two directions is a pattern field)

Let $U =$: the set of all squares of a doubly infinite tape, and let $\{L R\}$ satisfy: $L(x) =$: the square to the left of x ; $R(x) =$: the square to the right of x . Then, by definition 1, $F_2 = \langle U \{L R\} \rangle$ is a pattern field of dimension 2.

Example 1.3 (an infinite checkerboard is a pattern field)

Let $U =$: the set of all squares of an infinite checkerboard, and let $\{L R U D\}$ satisfy: $L(x) =$: the square to the left of x ; $R(x) =$: the square to the right of x ; $U(x) =$: the square above x ; $D(x) =$: the square below x . Then, by definition 1, $F_3 = \langle U \{L R U D\} \rangle$ is a pattern field of dimension 4.

I define a *generalized Turing machine* C on a pattern field F of dimension n as follows. Let $F = \langle U \{f_i\} \rangle$ and suppose a finite alphabet $A_C = \{a_i\}$ with at least one symbol, and a non-empty finite set $Q_C = \{q_i\}$ of internal states are given. A quadruple of the Turing machine C is of the two forms $q_r a_j a_i q_s$ or $q_r a_j f_i q_s$. A quadruple of the first form means: if the machine is in state q_r reading a_j , it will replace symbol a_j with symbol a_i , and it will change state to q_s . A quadruple of the

second form means: if the machine is in state q_r reading a_j , it will go to neighbor $f_i(x)$ (where $x \in U$ is the node where the machine is currently located), and it will change state to q_s .

Definition 2 (Turing machines on a pattern field)

A Turing machine C on a pattern field F of dimension n is individuated by a consistent set G_C of km quadruples, where k is the number of internal states in the state-set $Q_C = \{q_r\}$, m is the number of symbols in the alphabet $A_C = \{a_j\}$, and two quadruples are consistent just in case they are identical, or they do not begin with the same pair state/symbol¹².

I will consider the first symbol a_0 of an arbitrary alphabet $A = \{a_j\}$ a special symbol -- the null symbol, called the blank -- which I will also indicate by "b". A finite pattern (or more simply a pattern) of $F = \langle U \{f_i\} \rangle$ relative to alphabet $A = \{a_j\}$ is a function from U to A which assigns non-blank symbols to a *finite* number of nodes. I will indicate the set of all finite patterns of F relative to alphabet A by " $P_F(A)$ ".

Let C be a generalized Turing machine on pattern field $F = \langle U \{f_i\} \rangle$, and let $A_C = \{a_j\}$ and $Q_C = \{q_r\}$ be the alphabet and the state-set of C . Intuitively, a *total state* of a generalized Turing machine, specifies the information sufficient (and necessary) for determining the future behavior of the machine. Therefore, a total

¹² I have chosen this definition for technical reasons, namely, to make the total transition function of C (see below) a function which is defined for all total states of C . This definition is in fact equivalent to the usual one which allows any consistent set of quadruples. If such a set does not contain a quadruple beginning with $q_r a_j$, I simply add the quadruple $q_r a_j q_r$, and I adopt a slightly different convention to determine when a Turing machine stops. A machine stops iff its *total state* no longer changes (see below for the precise definition).

state is determined when a pattern $p \in P_F(A_C)$, an internal state $q_r \in Q_C$, and a node $x \in U$ are fixed. It is however useful to identify a total state with a special pattern. Suppose the total state of a Turing machine C is determined by pattern p , node x , and internal state q_r , and that the symbol on node x is a_j . Consider now the pattern w such that, on node x , w has the pair $q_r a_j$, and w agrees with p on all other nodes. Obviously, the total state determined by p , x , and q_r can be identified with this pattern $w \in P_F(B_C)$, where $B_C = A_C \cup \{q_r a_j\}$. Therefore, I formally define a total state as follows.

Let $Q_C = \{q_r\}$ and $A_C = \{a_j\}$ be, respectively, the state-set and the alphabet of a generalized Turing machine C on pattern field $F = \langle U \{f_j\} \rangle$. Let k be the number of internal states in Q_C and m be the number of symbols in A_C . Let B_C be the set which contains all symbols in A_C and all the pairs $q_r a_j$, where $q_r \in Q_C$ and $a_j \in A_C$. B_C has thus $m+km$ members. I will call this set the total alphabet of Turing machine C . Then, w is a total state of C iff $w: U \rightarrow B_C$ and there is exactly one node $x \in U$ such that $w(x) = q_r a_j$, for some $q_r \in Q_C$ and $a_j \in A_C$. We thus see that the *total states* of a Turing machine are a proper subset W_C of the set $P_F(B_C)$ of all finite patterns of F relative to the *total alphabet* of that machine. If, at time t , a Turing machine C is on node x of pattern $p \in P_F(A_C)$, and C is in internal state q_r , the total state of C at t is w such that $w(x) = q_r p(x)$ and, for all $y \neq x$, $w(y) = p(y)$.

If G_C are the quadruples of a generalized Turing machine C , G_C determines a function $g[G_C]: W_C \rightarrow W_C$, where $W_C \subseteq P_F(B_C)$ is the set of all total states of C ;

$g[G_C]$ is called the total transition function of C .

Once an initial pattern, an initial node, and an initial internal state are chosen, a generalized Turing machine C performs a *computation* on the pattern field, and C stops iff: for some time t , $g[G_C](w) = w$, where w is the total state of C at time t .

We can now define a computation of a function on pattern fields. Let C be a generalized Turing machine on pattern field F , and let $P_F(B_C)$ be the set of all patterns of F relative to the *total alphabet* B_C of C . Let $f: X \rightarrow Y$ be the function we want to compute. A coding of X to $P_F(B_C)$ is a function $\gamma: X \rightarrow P_F(B_C)$. A decoding of $P_F(B_C)$ to Y is a (partial) function¹³ $\delta: P_F(B_C) \rightarrow Y$. I then stipulate:

Definition 3 (computation of a function on a pattern field)

C computes f on F relative to γ and δ iff:

C is a generalized Turing machine on pattern field F , f is a (partial) function from X to Y , γ is a coding of X to $P_F(B_C)$, δ is a decoding of $P_F(B_C)$ to Y and, for any $x \in X$, if C is started on total state $\gamma(x)$ and $f(x)$ is defined, C stops and, if the final total state of C is w , $\delta(w)$ is defined and equal to $f(x)$. If $f(x)$ is undefined, C does not stop.

According to definition 3, if some generalized Turing machine computes f , f is a (partial) function whose domain and codomain are *arbitrary* sets. Furthermore, I have not imposed any restriction on the coding and decoding functions γ and δ . However, since we want to use generalized Turing machines to compute *numeric*

¹³ The reason why I allow a decoding to be a *partial* function is that, usually, decodings are only defined for *total states* of a Turing machine C , and the set W_C of all total states of C is a *proper* subset of $P_F(B_C)$.

functions¹⁴, we must introduce some *appropriate* convention for representing *numbers* as patterns.

If the pattern field satisfies some further conditions, there is a natural way of obtaining this representation. Suppose that, given a pattern field $F = \langle U \{f_i\} \rangle$, there is a Turing machine C on F , an internal state q_r of C , and a node $x \in U$ such that, when C is started on a *completely blank* pattern, node x , and internal state q_r , C will then visit all the nodes in U . If these conditions are satisfied, we can use this computation of the Turing machine to effectively enumerate all the nodes in U . In fact, let the number n ($0 \leq n$) correspond to the n -th step of such a computation. At each step, the machine is on exactly one node, and any node will be reached at some step. We thus obtain a (repetitive) enumeration of the nodes. We can then get a bijection $e: Z^+ \rightarrow U$ by first eliminating all the repetitions, and by then closing the gaps. I will call any bijection $e: Z^+ \rightarrow U$ which is obtained in this way an intrinsic enumeration¹⁵ of the pattern field $\langle U \{f_i\} \rangle$. A pattern field which has some intrinsic enumeration is called a regular pattern field. I give below two examples of regular pattern fields. In particular, I show that the tape of an ordinary

¹⁴ I will only consider numeric functions from Z^+ to Z^+ because any n -tuple of numbers can be recursively coded as one single number. More precisely, it can be proved that, for any n , there is a recursive bijection $\phi: Z^{+n} \rightarrow Z^+$. This bijection can thus be used to code any n -tuple $\langle m_1 \dots m_n \rangle$ as the number $\phi(m_1 \dots m_n)$. Furthermore, it can also be proved that there are n recursive functions $\phi_1 \dots \phi_n$ which satisfy $\phi_i(\phi(m_1 \dots m_n)) = m_i$ (Boolos and Jeffrey 1985, 161). These functions allow us to recover the i -th component of any n -tuple from its code. Therefore, for any (partial) function $g: Z^{+n} \rightarrow Z^{+k}$ the following equality holds: $g(m_1 \dots m_n) = \langle \phi_1(g_\phi(\phi(m_1 \dots m_n))) \dots \phi_k(g_\phi(\phi(m_1 \dots m_n))) \rangle$, where $g_\phi(\phi(m_1 \dots m_n)) =: \phi(g(m_1 \dots m_n))$. This means that any (partial) function g on n -tuples of numbers can always be identified with the corresponding function g_ϕ from Z^+ to Z^+ .

¹⁵ This enumeration is *intrinsic* in the sense that it depends on the neighborhood structure of the pattern field in an effective way.

Turing machine (infinite in one or two directions) is a regular pattern field.

Example 1.1.1 (a tape infinite in one direction is a regular pattern field)

Let $F_1 = \langle U \{L R\} \rangle$ be a tape infinite in one direction (see example 1.1). The Turing machine C_1 specified by the set of quadruples $G_1 = \{q_0 b R q_0\}$ reaches all the squares when started on the leftmost square of a completely blank tape. Therefore, F_1 is a regular pattern field.

Example 1.2.1 (a tape infinite in two directions is a regular pattern field)

Let $F_2 = \langle U \{L R\} \rangle$ be the doubly infinite tape of an ordinary Turing machine (see example 1.2). To show that F_2 is a regular pattern field, we must exhibit a Turing machine C_2 which, when started on a completely blank tape, reaches all the squares. Let $A = \{b 1\}$ be the alphabet of C_2 . C_2 is individuated by the set of quadruples $G_2 = \{q_0 b 1 q_1, q_1 1 L q_1, q_1 b 1 q_0, q_0 1 R q_0\}$. When C_2 is started on a blank tape in state q_0 , it first writes a 1 and changes state to q_1 . Then, it goes to the first square to the left, where it writes a second 1, and changes state to q_0 . Now, it moves to the right skipping all the ones, and as soon as it finds a blank, writes a 1. Then, it moves to the left skipping the ones, replaces the first blank with a 1, and repeats. Therefore, C_2 visits all the squares of the tape. C_2 thus determines an intrinsic enumeration of the tape F_2 . This enumeration assigns the number 0 to the square where C_2 starts, increasing even numbers to the squares to its right, and increasing odd numbers to the squares to its left. I will call this intrinsic enumeration the standard enumeration of the tape.

Consider now a regular pattern field $F = \langle U \{f_i\} \rangle$, and let $e: Z^+ \rightarrow U$ be an intrinsic enumeration of F . This bijection transfers the natural ordering of the non-negative integers to the set of nodes U . It thus makes sense to talk of the first, the second, the third, ... node of U . Let $P_F(A)$ be the set of all finite patterns of F relative to alphabet $A = \{a_i\}$, where A has m elements. Once e is *fixed*, each finite pattern p is a numeral in base m , whose least significant digit is the symbol on the first node of U relative to e , and whose highest digit is the symbol on the

last non-blank node according to the order induced by e . Therefore, when e is fixed, each finite pattern p denotes a number n . I make explicit this fact by indicating a finite pattern p by " n_e ", where n is the number denoted by p when the intrinsic enumeration e of F is specified.

We are now in the position of *effectively coding* numbers as patterns. In fact, since e is fixed, each finite pattern p can be identified with the number n such that $p = n_e$. Therefore, an *effective coding of numbers as patterns* reduces to an *effective rule* which, to any *number*, assigns another *number*. Now, such an effective rule can obviously be identified with a recursive function. Let F be a regular pattern field with a fixed intrinsic enumeration e , and let C be a generalized Turing machine on F with *total alphabet* B_C . Let $P_F(B_C)$ be the set of all finite patterns of F relative to B_C . Then, I define:

Definition 4 (coding numbers as finite patterns)

c is a coding of Z^+ to $P_F(B_C)$ relative to e iff:
 c is a coding of Z^+ to $P_F(B_C)$, and there is a recursive function c_e such that, if $c(k) = n_e$, then $c_e(k) = n$.

Example 4.1 (the usual conventions for representing input numbers as total states of ordinary Turing machines specify a coding relative to the standard enumeration of the tape)

Suppose C is an ordinary Turing machine with alphabet $A_C = \{a_i\} = \{0\ 1\}$, where 0 is the blank. Input number k is usually represented on the tape by means of a block of $k+1$ 1s. Let e be the standard enumeration of the tape (see example 1.2.1), and let x be the first square of the tape relative to the ordering induced by e . I assume that the block of $n+1$ 1s which represents number n has its leftmost 1 on square x . This convention thus individuates a function $\pi: Z^+ \rightarrow P_F(A_C)$, where $P_F(A_C)$ is the set of all finite patterns relative to the alphabet $A_C = \{0\ 1\}$. I now verify that there is a

recursive function π_e such that, if $\pi(k) = n_e$, then $\pi_e(k) = n$.

The standard enumeration e assigns the number 0 to the square x , increasing even numbers to the squares to its right, and increasing odd numbers to the squares to its left (see how the Turing machine C_2 of example 1.2.1 works).

Therefore, π assigns to number k the finite pattern $\pi(k) = n_e$ which denotes the number $n = 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + \dots + 0 \cdot 2^{2k-1} + 1 \cdot 2^{2k}$. Let then $\pi_e: Z^+ \rightarrow Z^+$ satisfy:

$$\pi_e(k) =: 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + \dots + 0 \cdot 2^{2k-1} + 1 \cdot 2^{2k};$$

by its definition, π_e is recursive and, if $\pi(k) = n_e$, $\pi_e(k) = n$.

The usual start-convention $\sigma: P_F(A_C) \rightarrow P_F(B_C)$ for an ordinary Turing machine consists in letting the machine start in state q_0 on the leftmost 1 of the tape, if such a 1 exists. Otherwise, σ is undefined. Suppose the state-set $Q_C = \{q_j\}$ of Turing machine C has s ($0 \leq j \leq s-1$) elements, and let the elements of the total alphabet $B_C = \{a_i\} \cup \{q_j a_i\}$ be alphabetically ordered¹⁶. We have just seen that the initial pattern n_e is a numeral in base two which denotes the number:

$$n = 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + \dots + 0 \cdot 2^{2k-1} + 1 \cdot 2^{2k};$$

on the other hand, the initial total state $\sigma(n_e) = m_e$ determined by the start-convention σ is a numeral in base $2+s2$ which denotes the number:

$$m = (2+1) \cdot (2+s2)^0 + 0 \cdot (2+s2)^1 + 1 \cdot (2+s2)^2 + 0 \cdot (2+s2)^3 + \dots + 0 \cdot (2+s2)^{2k-1} + 1 \cdot (2+s2)^{2k};$$

let $\sigma_e(n) =: m$, if the expression in base two of n is of the form $1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + \dots + 0 \cdot 2^{2k-1} + 1 \cdot 2^{2k}$; otherwise, $\sigma_e(n) =: n$. Then, by its definition, σ_e is recursive and, if $\sigma(n_e) = m_e$, $\sigma_e(n) = m$.

Finally, let us define $c: Z^+ \rightarrow P_F(B_C)$ by $c(k) =: \sigma(\pi(k))$. Then, by definition 4, c is a coding of Z^+ to $P_F(B_C)$ relative to e .

The inverse problem of retrieving, or *decoding*, a number from a finite pattern can easily be solved in an analogous manner. When an intrinsic enumeration is fixed, decoding a number from a pattern reduces to specifying an *effective* rule which, to the number denoted by the pattern, associates another number. Therefore, we can identify such a rule with a recursive function. I thus define:

¹⁶ Since B_C is alphabetically ordered, the symbol a_i ($0 \leq i < 2$) corresponds to number i , and the pair $q_0 a_i$ corresponds to number $2+i$.

Definition 5 (decoding numbers from finite patterns)

d is a decoding of $P_F(B_C)$ to Z^+ relative to e iff:

d is a decoding of $P_F(B_C)$ to Z^+ , and there is a recursive function d_e such that, if $d(n_e) = k$, then $d_e(n) = k$.

Example 5.1 (the usual conventions for retrieving output numbers from total states of ordinary Turing machines specify a decoding relative to the standard enumeration of the tape)

Suppose C is an ordinary Turing machine with alphabet $A_C = \{a_i\} = \{0, 1\}$, where 0 is the blank. The usual convention for retrieving output numbers is that a tape with n 1s represents number n . This rule thus determines a function $\eta: P_F(A_C) \rightarrow Z^+$, where $P_F(A_C)$ is the set of all finite patterns relative to the alphabet $A_C = \{0, 1\}$. Let e be the standard enumeration of the tape (see example 1.2.1). I now verify that there is a recursive function η_e such that, if $\eta(n_e) = k$, then $\eta_e(n) = k$.

Since e is fixed, any finite pattern n_e is a numeral in base two which denotes number n and, by the definition of η , η assigns to pattern n_e the number of 1s in the binary expression of n . Let then $\eta_e: Z^+ \rightarrow Z^+$ satisfy:

$\eta_e(n) =$: the number of 1s in the binary expression of n ;

by its definition, η_e is recursive and, if $\eta(n_e) = k$, $\eta_e(n) = k$.

Suppose the state-set $Q_C = \{q_j\}$ of Turing machine C has s ($0 \leq j \leq s-1$) elements, and let the elements of the total alphabet $B_C = \{a_i\} \cup \{q_j a_i\}$ be alphabetically ordered¹⁷. Let $\phi: P_F(B_C) \rightarrow P_F(A_C)$ be the partial function which, to any total state $w \in W_C \subset P_F(B_C)$, assigns the pattern $p \in P_F(A_C)$ such that $p(x) = w(x)$, if $w(x) \neq q_j a_i$, otherwise, $p(x) = a_i$. Since the standard enumeration of the tape e is fixed, an arbitrary total state w of C is a numeral m_e in base $2+s$ which denotes the number

$$m = c_0 \cdot (2+s)^0 + c_1 \cdot (2+s)^1 + \dots + c_u \cdot (2+s)^u,$$

where exactly one coefficient $c_s = 2+j2+i$, for some j ($0 \leq j < s$) and i ($0 \leq i < 2$). On the other hand, the pattern $p = \phi(m_e)$ is a numeral n_e in base two which denotes the number

$$n = d_0 \cdot 2^0 + d_1 \cdot 2^1 + \dots + d_u \cdot 2^u,$$

where all the coefficients agree with the corresponding coefficients of m , except for $d_s = i$. Let $\phi_e: Z^+ \rightarrow Z^+$ satisfy:

$\phi_e(m) = n$, if m is a number denoted by some total state m_e ; $\phi_e(m) = m$, otherwise;

¹⁷ Since B_C is alphabetically ordered, the symbol a_i ($0 \leq i \leq 1$) corresponds to number i , and the pair $q_j a_i$ ($0 \leq j \leq s$) corresponds to number $2+j2+i$.

then, ϕ_e is recursive¹⁸ and, if $\phi(m_e) = n_e$, then $\phi_e(m) = n$.

Finally, let $d: P_F(B_C) \rightarrow Z^+$ be the partial function such that $d(m_e) = \eta(\phi(m_e))$ if m_e is a total state of C ; undefined otherwise. Then, by definition 5, d is a decoding of $P_F(B_C)$ to Z^+ relative to e .

For *regular* pattern fields with a *fixed* intrinsic enumeration e , we can finally stipulate that a numeric function is COMPUTABLE just in case a Turing machine computes that function relative to some numeric coding c and decoding d . I give below the formal definition:

Definition 6 (numeric COMPUTABLE functions on a regular pattern field with a fixed intrinsic enumeration)

f is a COMPUTABLE function on F relative to e iff:

f is a (partial) function from Z^+ to Z^+ , there is a generalized Turing machine C on F , and there are c and d such that c is a coding of Z^+ to $P_F(B_C)$ relative to e , d is a decoding of $P_F(B_C)$ to Z^+ relative to e , and C computes f on F relative to c and d .

I prove below that any numeric function which is computable by an ordinary Turing machine with the usual conventions satisfies definition 7 (when e is identified with the standard enumeration of the tape).

Example 6.1 (all the numeric functions computable by an ordinary Turing machine with the usual conventions are COMPUTABLE relative to the standard enumeration of the tape)

Let $F_2 = \langle U \{L R\} \rangle$ be the doubly infinite tape of an ordinary Turing machine (see example 1.2). Let e be the intrinsic enumeration of F_2 determined by the Turing

¹⁸ m is a number denoted by some total state just in case its decomposition in base $2+s_2$ has exactly one coefficient $c_s = 2+j_2+i$, for some j ($0 \leq j < s$) and i ($0 \leq i < 2$). This condition is recursive. Therefore, ϕ_e is *recursively* defined by cases, and it is thus recursive.

machine C_2 of example 1.2.1. Then, all the numeric functions computable by an ordinary Turing machine with the usual conventions are COMPUTABLE on F_2 relative to e . Let c be the coding of Z^+ to $P_F(B_C)$ relative to e of example 4.1, and d be the decoding of $P_F(B_C)$ to Z^+ relative to e of example 5.1. The usual definition of a numeric Turing computable function is the following:

A (partial) function f from Z^+ to Z^+ is Turing computable iff:
there is a Turing machine C such that, when C is started in state q_0 , on the leftmost 1 of a block of $n+1$ 1s, C stops with $f(n)$ 1s on the tape, if $f(n)$ is defined. If $f(n)$ is not defined, C does not stop.

By the previous definition, and by definition 3, if f is Turing computable, there is a Turing machine C such that C computes f on F_2 relative to c and d . Since c and d are, respectively, a coding of Z^+ to $P_F(B_C)$ relative to e , and a decoding of $P_F(B_C)$ to Z^+ relative to e , by definition 6, f is COMPUTABLE on F_2 relative to e q.e.d.

3. The relation between generalized Turing machines and ordinary ones

We have seen in example 1.2.1 that the tape of an ordinary Turing machine is a regular pattern field. Therefore, these devices are a special type of generalized Turing machines. We have also seen that all the numeric functions computable by an ordinary Turing machine with the usual conventions are COMPUTABLE functions relative to the standard enumeration of the tape (see example 6.1). It is now natural to consider the converse question: are all the numeric functions which can be computed by a generalized Turing machine also computable by an ordinary one? If we believe in Turing's *fourth hypothesis* (see sec. 1), we must conclude that generalized Turing machines cannot be more powerful than ordinary ones. However, before accepting this conclusion, we should take a closer look to the capabilities of a generalized Turing machine which depend on the neighborhood structure of its pattern field. The pattern field of ordinary Turing machines has a very simple relational structure. But what happens if the connections between the nodes are more complicated? Could a more complex neighborhood structure allow a generalized Turing machine to compute non-recursive functions?

We will see below that the surprising answer to this question is affirmative. There are generalized Turing machines which can compute non-recursive functions. In fact, if we appropriately choose the neighbor functions, any numeric function turns out to be computable on some pattern field. This result crucially depends on the fact that some pattern fields have a neighborhood structure

essentially more complex than the structure of the tape of an ordinary Turing machine. The basic intuition can be expressed as follows. Given a doubly infinite tape with the standard enumeration of its squares, the neighbor functions Left and Right individuate two numeric *recursive* functions¹⁹. However, if a more complicated pattern field is given, there may be an intrinsic enumeration which allows us to represent the neighbor functions as non-recursive functions. Now, if this is the case, this non-recursive power embedded in the pattern field may be used by a Turing machine which operates on it to compute non-recursive functions. Before proving these results, however, we need to precisely define the idea of a pattern field whose neighborhood structure is recursive. Intuitively, when an intrinsic enumeration is *fixed*, a pattern field is recursive if all the neighbor functions turn out to be recursive with respect to that enumeration. I formally define this concept as follows²⁰:

¹⁹ I believe that, for *any* intrinsic enumeration of a doubly infinite tape, the functions Left and Right individuate recursive functions. However, I have not been able to prove this conjecture. This proof would be important because it would show that there is no intrinsic way of representing the neighbor functions of an ordinary tape in a non-recursive way. Therefore, no "non-recursive power" would be hidden in the infinite tape of ordinary Turing machines, and this would explain why such devices cannot compute non-recursive functions. In fact, I will prove the following proposition (sec. 4, cor. 5.1): if a pattern field F is recursive relative to an intrinsic enumeration e , and f is COMPUTABLE on F relative to e , then f is (partial) recursive. Consequently, if the tape of ordinary Turing machines is recursive relative to *any* intrinsic enumeration, it is *absolutely* impossible for these machines to COMPUTE non-recursive functions.

²⁰ I believe that this definition does not depend on the intrinsic enumeration e that we choose. However, I have not been able to prove this conjecture. More precisely, the following hypothesis is likely to be true: if a pattern field F is recursive relative to e , then F is recursive relative to any other intrinsic enumeration $e^\#$. If this is true, then being a recursive (or non-recursive) pattern field is an intrinsic property of the pattern field. In other words, no recursive pattern field has a non-recursive power, and all non-recursive pattern fields have a non-recursive power relative to any intrinsic enumeration e .

Definition 7 (recursive pattern field relative to an intrinsic enumeration)

A pattern field $F = \langle U \{f_i\} \rangle$ is recursive relative to e iff:

F is regular, $e: Z^+ \rightarrow U$ is an intrinsic enumeration of F and, for any i , $f_{e_i}: Z^+ \rightarrow Z^+$ is recursive, where $f_{e_i}(n) =: e^{-1}(f_i(e(n)))$.

I prove below that the tape of an ordinary Turing machine is a recursive pattern field relative to its standard enumeration, and I then give a simple example of a regular pattern field which is not recursive.

Example 7.1 (the tape of an ordinary Turing machine is recursive relative to the standard enumeration)

We have seen in example 1.2.1 that a doubly infinite tape $F = \langle U \{L, R\} \rangle$ is a regular pattern field. Let e be the standard enumeration of F . The numeric functions L_e and R_e which, respectively, correspond to L and R under e are defined by:

$L_e(n) =: 1$, if $n = 0$; $n+2$, if n is odd; $n-2$, if n is even

$R_e(n) =: n+2$, if $n = 0$ or n is even; 0 , if $n = 1$; $n-2$, if n is odd and $n \neq 1$

by their definitions, L_e and R_e are recursive, whence, by definition 7, F is recursive relative to e .

Example 7.2 (a regular pattern field which is not recursive)

Let $F = \langle Z^+ \{s, f\} \rangle$, where Z^+ are the non-negative integers, s is the successor function, and f is a non-recursive function. Then, the identity function on Z^+ , e , is an intrinsic enumeration of F , for it is individuated by the Turing machine C whose set of quadruples is $G = \{q_0, b, s, q_0\}$. In fact, when C is started on node 0 of a completely blank pattern, C reaches all the nodes. Therefore, the function which corresponds to f under e is f itself, whence F is not recursive relative to e .

I am now going to prove two general results. The first theorem states that there is a pattern field F , and an intrinsic enumeration e , such that F is recursive

relative to e , and all the (partial) recursive functions are COMPUTABLE on F relative to e . In fact, this pattern field is the tape of an ordinary Turing machine, and e is the standard enumeration of the tape. This theorem thus is an immediate consequence of examples 6.1 and 7.1.

Theorem 1 (there is a recursive pattern field on which all the (partial) recursive functions are COMPUTABLE relative to an intrinsic enumeration)

There is a pattern field F , and an intrinsic enumeration e of F such that F is recursive relative to e and, if $f: \mathbb{Z}^+ \rightarrow \mathbb{Z}^+$ is (partial) recursive, f is COMPUTABLE on F relative to e

proof:

let F be the doubly infinite tape of an ordinary Turing machine, and e be the standard enumeration of the tape. Then, the thesis follows from examples 7.1, 6.1, and from the fact that all (partial) recursive functions are computable by an ordinary Turing machine with the usual conventions q.e.d.

The second theorem states that, for any function f which is not (partial) recursive, there is a pattern field F , and an intrinsic enumeration e , such that F is not recursive relative to e , and f is COMPUTABLE on F relative to e . In other words, given any non-recursive function f , it is always possible to find a pattern field on which this function can be computed. However, the neighborhood structure of this pattern field turns out to be non-recursive as well. The proof of this theorem is based on a very simple idea. The trick consists in letting the nodes of the pattern field to be the integers themselves, so that f (the function to be computed) can be taken to be one of the neighbor functions. We then need to add two more neighbor functions, the successor function s , and the predecessor

function v . This is sufficient for proving the theorem when f is total. In fact, f can be computed on the pattern field $\langle Z^+ \{s \ v \ f\} \rangle$, where Z^+ are the non-negative integers, s is the successor function, and v is the predecessor function.

If f is partial, we need to consider the total function ϕ such that, when $n \neq 0$, $\phi(n) = f(n-1) + 1$, if $f(n-1)$ is defined; if $f(n-1)$ is undefined, $\phi(n) = 0$. When $n = 0$, $\phi(n) = 0$. For example, suppose $f(0) = 3$, $f(1) = \text{undefined}$, $f(2) = 123$, $f(3) = 0$, etc. Then, $\phi(0) = 0$, $\phi(1) = 4$, $\phi(2) = 0$, $\phi(3) = 124$, $\phi(4) = 1$, etc. Notice that, since f is partial, but not partial recursive, ϕ is total but not recursive²¹. The pattern field on which f turns out to be computable is $\langle Z^+ \{s \ v \ \phi\} \rangle$, where Z^+ are the non-negative integers, s is the successor function, and v the predecessor function.

Theorem 2 (any numeric function which is not (partial) recursive is COMPUTABLE on some non-recursive pattern field)

If $f: Z^+ \dashrightarrow Z^+$ is not (partial) recursive, there is a pattern field F , and an intrinsic enumeration e of F such that F is not recursive relative to e , and f is COMPUTABLE on F relative to e

proof:

I divide the proof in two cases: (a) f is total but not recursive; (b) f is partial but not partial recursive

proof of (a):

let $F =: \langle Z^+ \{s \ v \ f\} \rangle$, where Z^+ are the non-negative integers, s is the successor function, and v is the predecessor function (where $v(0) =: 0$). Let e be the identity function on Z^+ . By example 7.2, e is an intrinsic enumeration of F . Since f is not recursive, and e is the identity function on Z^+ , F is not recursive relative to e . I now show that f is COMPUTABLE on F relative to e . Let $P_F(A_C)$ be the set of all finite

²¹ If ϕ were recursive, ϕ would be computable by an ordinary Turing machine $C(\phi)$. Then, we could construct a second ordinary Turing machine $C(f)$ which computes f . To obtain $C(f)$, we add a routine which, whenever, $C(\phi)$ stops, checks whether the output number is 0. If yes, $C(f)$, enters an infinite loop. If not, $C(f)$ stops. Since f is computable by an ordinary Turing machine, f is partial recursive, against the hypotheses.

patterns of F relative to alphabet $A_C = \{b, a_1\}$. Let us represent input (output) number n as a block of $n+1$ a_1 's with the first a_1 on node 0, the second a_1 on node 1 etc. Consider now the Turing machine C whose states are $Q_C = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$, and whose set of quadruples is $G_C = \{q_0 a_1 s q_1, q_1 a_1 b q_2, q_2 b s q_1, q_1 b v q_3, q_3 b f q_4, q_3 a_1 f q_4, q_4 b a_1 q_5, q_5 a_1 v q_4, q_4 a_1 a_1 q_6\}$. If we start C in state q_0 , on node 0, and on a block of $n+1$ a_1 's, C stops in state q_6 scanning a_1 , and the output pattern is a block on $f(n)+1$ a_1 's. In fact, C first leaves the marker a_1 on node 0, then deletes all other a_1 's. As soon as C finds the first blank, it goes one node back, and then it jumps according to f . Now the pattern is completely blank except for node 0 where there is a_1 , and the machine is on node $f(n)$. The machine now goes back writing a_1 on each blank node until it finds the a_1 on node 0. At that point, it stops in state q_5 . Define: $\pi: Z^+ \rightarrow P_F(A_C)$, $\pi(n) =$: a block of $n+1$ a_1 's with the first a_1 on node 0, the second a_1 on node 1 etc; $\sigma: P_F(A_C) \rightarrow P_F(B_C)$, $\sigma(p) =$: $w \in W_C$ such that $w(k) = q_0 p(k)$, if k is node 0; $w(k) = p(k)$, otherwise; $c: Z^+ \rightarrow P_F(B_C)$, $c(n) =$: $\sigma(\pi(n))$; $\phi: P_F(B_C) \rightarrow P_F(A_C)$ be the partial function which, to any total state $w \in W_C \subset P_F(B_C)$, assigns the pattern $p \in P_F(A_C)$ such that $p(k) = w(k)$, if $w(k) \neq q_3 a_1$, otherwise, $p(k) = a_1$; $\eta: P_F(A_C) \rightarrow Z^+$, $\eta(p) =$: $n - 1$, if $n \neq 0$, otherwise, $\eta(p) =$: n , where n is the number of a_1 's contained in pattern p ; $d: P_F(B_C) \rightarrow Z^+$ be the partial function such that $d(m_e) = \eta(\phi(m_e))$. Then, c is a coding of Z^+ to $P_F(B_C)$ relative to e (the proof is similar to example 4.1), and d is a decoding of $P_F(B_C)$ to Z^+ relative to e (the proof is similar to example 5.1). Furthermore, C computes f on F relative to c and d . Therefore, by def. 6, f is COMPUTABLE on F relative to e //q.e.d.//

proof of (b):

let $F = \langle Z^+ \{s \vee \phi\} \rangle$, where Z^+ are the non-negative integers, s is the successor function, v is the predecessor function (where $v(0) = 0$), and ϕ is the function defined in the paragraph above the statement of theorem 2. Let e be the identity function on Z^+ . By example 7.2, e is an intrinsic enumeration of F . Since ϕ is not recursive (see footnote 21), and e is the identity function on Z^+ , F is not recursive relative to e . I now show that f is COMPUTABLE on F relative to e . Let $P_F(A_C)$ be the set of all finite patterns of F relative to alphabet $A_C = \{b, a_1\}$. Let us represent input (output) number n as a block of $n+1$ a_1 's with the first a_1 on node 0, the second a_1 on node 1 etc. Let c and d be defined as in proof (a). Consider now the Turing machine C whose states are $Q_C = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$, and whose set of quadruples is $G_C = \{q_0 a_1 s q_1, q_1 a_1 b q_2, q_2 b s q_1, q_1 b \phi q_3, q_3 a_1 a_1 q_4, q_4 a_1 a_1 q_3, q_3 b v q_5, q_5 b a_1 q_6, q_6 a_1 v q_5, q_5 a_1 a_1 q_7\}$. If we start C in state q_0 , on node 0, and on a block of $n+1$ a_1 's, C stops in state q_7 scanning a_1 , and the output pattern is a block on $f(n)+1$ a_1 's, if $f(n)$ is defined. Otherwise, C does not stop. In fact, C first leaves the marker a_1 on node 0, then deletes all other a_1 's. As soon as C finds the first blank, it jumps according to ϕ . Now the pattern is completely blank except for node 0 where there is a_1 , and the machine is on node $f(n)+1$, if $f(n)$ is defined; otherwise it is on node 0 scanning a_1 . The machine now checks whether it is scanning a_1 . If yes, it enters an infinite loop (it stays indefinitely on node 0 scanning a_1 and changing state from q_3 to q_4). If not, the machine first goes one node back, and it then writes a_1 on each blank node, going back until it finds the a_1 on node 0. At that point, it stops in state q_7 . Therefore, C computes f on F relative to c and d . From this, the definitions of c and d , and def. 6, f is COMPUTABLE on F relative

to e. //q.e.d.// Theorem 2 is thus proved²² q.e.d.

It is interesting to contrast the way a *generalized Turing machine* can compute a non-recursive function with the way an *oracle machine* can achieve the same result. Oracle machines are ordinary Turing machines equipped with a special device (oracle) which always gives the right answer to questions of the form: is number n a member of set X ? The set X is a fixed, arbitrary, subset of the non-negative integers. In the course of a computation an oracle machine can pause and ask its oracle whether the number of 1s on the tape at that moment is a member of X . The machine then resumes the computation by choosing between two alternatives according to the answer of the oracle. Davis has proved²³ that oracle machines can compute all (and only) the numeric functions which are obtained from a set of basic functions by applying a finite number of times the operations of primitive recursion, composition, and minimization. The basic functions are the usual basic recursive functions (successor, zero function, and identity functions) and the characteristic function f_x of X . Oracle machines can

²² Notice that, since $\langle Z^+ \{s v\} \rangle$ is isomorphic to a tape infinite in one direction, the proof of Theorem 2 tells us how to enhance the computational power of ordinary Turing machines. If we want to compute a total non-recursive numeric function f_e , we need to add a neighbor function f which connects the n -th ($0 \leq n$) square to the $f_e(n)$ -th square. An analogous trick can easily be found for doubly infinite tapes: the neighbor function f should now connect the $2 \cdot n$ -th ($0 \leq n$) square to the $2 \cdot f_e(n)$ -th square (where squares are counted according to the standard enumeration of the tape e). If f_e is partial, and the tape is infinite in one direction, we use the corresponding total function ϕ_e , so that we add the neighbor function ϕ which connects the $(n+1)$ -st square to the $(f_e(n)+1)$ -st square. If x is the first square of the tape, $\phi(x) = x$ and, if $f_e(n)$ is undefined, ϕ connects the $(n+1)$ -st square to x . Analogously, if the tape is doubly infinite.

²³ Davis (1958). See ch. 1, sec. 4, ch. 2, ch. 3, sec. 1, and ch. 4, sec. 2, cor. 2.3.

thus compute non-recursive functions if X is a non-recursive set.

The main difference between oracle machines and generalized Turing machines consists in where the non-recursive power is located. The non-recursive power of an oracle machine is in its oracle. The non-recursive power of a generalized Turing machine, instead, is in the neighborhood structure of its pattern field. Nevertheless, we can find a class of generalized Turing machines which corresponds to an arbitrary class of oracle machines. Consider all the machines equipped with an oracle which computes the characteristic function f_X of a fixed subset X of the non-negative integers. The generalized Turing machines which correspond to this class of oracle machines are then all those machines which operate on the pattern field $\langle Z^+ \{s\} \cup \{f_X\} \rangle$, where Z^+ are the non-negative integers, s is the successor function and v is the predecessor function.

This class of generalized Turing machines has exactly the same computational power as the corresponding class of oracle machines. Let $\{f_i\}$ be a finite set of numeric functions. Then, g is (partial) recursive relative to $\{f_i\}$ just in case g is a (partial) numeric function which can be obtained from $\{f_i\}$ and the set of basic recursive functions (successor, zero function, and identity functions) by applying a finite number of times the operations of primitive recursion, composition, and minimization. Oracle machines whose oracles compute the characteristic function f_X of $X \subseteq Z^+$ can compute all and only the (partial) recursive functions relative to $\{f_X\}$. Generalized Turing machines which operate on $\langle Z^+ \{s\} \cup \{f_X\} \rangle$ can compute exactly the same class of numeric functions. In fact, I will prove below that all the

(partial) recursive functions relative to $\{f_i\}$ are COMPUTABLE on $\langle Z^+ \{s v\} \cup \{f_i\} \rangle$ relative to the identity function e on Z^+ . That only these numeric functions are COMPUTABLE on $\langle Z^+ \{s v\} \cup \{f_i\} \rangle$ relative to e will then follow from a more general theorem which I will prove in the next section (th. 5).

Theorem 3 (the class of the (partial) recursive functions relative to $\{f_i\}$ is included in the class of the functions COMPUTABLE on $\langle Z^+ \{s v\} \cup \{f_i\} \rangle$ relative to the identity function on Z^+)

If $g: Z^+ \rightarrow Z^+$ is a (partial) recursive function relative to $\{f_i\}$ then g is COMPUTABLE on $F = \langle Z^+ \{s v\} \cup \{f_i\} \rangle$ relative to the identity function e on Z^+

proof:

in the first place, notice that each node $k \in Z^+$ can be identified with the square of a tape infinite in one direction, and that s and v thus correspond, respectively, to the *Right* and *Left* functions of ordinary Turing machines which operate on this kind of tape. Suppose that we represent input (output) number x as a block of $x+1$ a_i s with the first a_1 on square 0, the second a_1 on square 1 etc. If the input numbers are more than one, each of them is represented by the appropriate block of a_i s, and different blocks are separated by a blank b . Let us also assume that generalized Turing machines which operate on F are always started on square 0, and in state q_0 . Furthermore, they always stop on square 0. Let us call any numeric function of m arguments which can be computed on F according to these conventions a *standard computable function on F*. It is then clear that any standard computable function of one argument is COMPUTABLE on F relative to the identity function e on Z^+ . Therefore, I must only show that all (partial) recursive functions relative to $\{f_i\}$ are standard computable on F .

This can be obtained by slightly modifying Boolos and Jeffrey's proof that all (partial) recursive functions are Turing computable (1980, ch. 6 and ch. 7). Boolos and Jeffrey first prove that all abacus computable functions are Turing computable (ch. 6), and then prove that all (partial) recursive functions are abacus computable (ch. 7).

An abacus is a machine with an infinite number of registers $[1] [2] \dots [k] \dots$. Each register can hold an arbitrary non-negative integer, and the machine performs just two types of operations on the registers: $\rightarrow 1_+[k]$ and $\rightarrow 1_+[k]-_1/-_2$. The $1_+[k]$ operation goes to register $[k]$, adds 1 to the number in this register, and then exits along path \rightarrow . The $1_+[k]-_1/-_2$ operation goes to register $[k]$ and checks whether the number x in this register is different from zero. If $x \neq 0$, it subtracts 1 from x , and then exits along path $-_1$. Otherwise, it simply exits along path $-_2$. Any abacus is specified by a *program*, that is, a flow graph which indicates the sequence in which these two types of operations are to be performed. A (partial) numeric function f is abacus computable just in case there is an abacus such that, when it is started with the

arguments $x_1 \dots x_m$ of the function in registers $[0] \dots [m]$, and with all other registers empty, it stops and the content of a specified register $[n]$ is equal to $f(x_1 \dots x_m)$ if $f(x_1 \dots x_m)$ is defined. Otherwise, the abacus does not stop. It is then easy to prove that all (partial) recursive functions are abacus computable (see Boolos and Jeffrey 1980, ch. 7).

In fact, abaci can be modified so that they are able to compute all (partial) recursive functions relative to $\{f_i\}$. This can simply be obtained by adding the operation $\rightarrow f_i[k] \rightarrow$, for any i . If the number in register $[k]$ is x , this operation goes to that register, changes x to $f_i(x)$, and then exits along path \rightarrow . Let us call any abacus with these special operations an $\{f_i\}$ -abacus. Then, it is obvious that all (partial) recursive functions relative to $\{f_i\}$ are $\{f_i\}$ -abacus computable (the proof is exactly the same as the one for (partial) recursive functions).

According to Boolos and Jeffrey, a (partial) numeric function f is Turing computable just in case there is a Turing machine with alphabet $\{b a_1\}$ which computes f according to the following conventions: (1) the tape is infinite in only one direction, and input (output) number x is represented by a block of $x+1$ a_1 s. The leftmost a_1 is on the third square of the tape. If there are more than one input numbers, each of them is represented by the appropriate block of a_1 s, and different blocks are separated by exactly one blank b ; (2) the Turing machine always starts on the third square of the tape, and stops on the same square. Boolos and Jeffrey then show how an arbitrary program of an abacus which computes a function f can be transformed into a table of a Turing machine which computes the same function according to conventions (1) and (2). Abacus programs are flow graphs with just two types of operations: $\rightarrow 1_+[k] \rightarrow$ and $\rightarrow 1_+[k] \rightarrow -_1 -_2$. Boolos and Jeffrey give the tables of two Turing machines with alphabet $\{b a_1\}$ which correspond to these two types of operation. In broadest outline, these machines work as follows. The number x in register $[k]$ is represented on the tape by a block of $x+1$ a_1 s. This block is the k -th block of a_1 s on the tape, blocks are separated by blanks, and the leftmost a_1 of the first block is on the third square of the tape. The machine which corresponds to the $1_+[k]$ operation starts on the leftmost a_1 (that is on the third square), and then goes to the right until it finds the k -th block. It then writes a_1 at the end of this block, shifts all the remaining blocks one square to the right, and then goes back to the third square and stops. The machine which corresponds to the $1_+[k]$ operation starts on the leftmost a_1 (on the third square), and then goes to the right until it finds the k -th block. It then checks whether this block only contains one a_1 . If yes, it goes back to the third square and stops. Otherwise, it deletes the rightmost a_1 of the k -th block, shifts all the remaining blocks one square to the left, and then goes back to the third square and stops. It is thus clear that, by appropriately combining these two types of machines, the program of an abacus which computes function f can be transformed into the table of a Turing machines which computes the same function according to conventions (1) and (2).

I am now going to modify this proof in order to show that any program of an $\{f_i\}$ -abacus which computes function f can be transformed into the table of a generalized Turing machine on F which computes the same function according to conventions (1) and (2). This obviously implies that f is standard computable on F . Therefore, since all (partial) recursive functions relative to $\{f_i\}$ are $\{f_i\}$ -abacus computable, all these functions are standard computable on F , and theorem 3 is thus

proved.

The only difference between an $\{f_i\}$ -abacus program and a normal abacus program is the presence of nodes of type $\rightarrow f_i[k] \rightarrow$. Therefore, I must only describe a generalized Turing machine C_i which corresponds to this type of node. Assume that the number x in register k is represented on the tape by the k -th block of a_1 's, and that this block has $x+1$ a_1 's. C_i has alphabet $\{b, a_1, ba_1, a_1a_1, \#_1, \#_2, \dots, \#_u\}$ and it starts on the leftmost a_1 (on the third square). It then goes to the right until it finds the k -th block of a_1 's. Then, C_i makes a special copy of this block. Recall that the k -th block has $x+1$ a_1 's. C_i writes a copy of the k -th block on the first $x+1$ squares of the tape. This copy, however, is written without deleting the content of these squares. This is obtained by using the special symbols " ba_1 " and " a_1a_1 ". That is, if square y ($0 \leq y < x+1$) contains symbol " b ", then C_i replaces " b " with " ba_1 "; if square y contains symbol " a_1 ", then C_i replaces " a_1 " with " a_1a_1 ". To keep track of the progress of the copying process C_i may use some of the markers $\#_1, \#_2, \dots, \#_u$. All the markers, however, are deleted when the copying is complete and, at the end of this process, C_i is on the first square of the tape.

At this point, C_i starts a second routine which computes the function f_i . The quadruples G_i of this subroutine are obtained from the quadruples G_C given in **proof (a)** of theorem 2, by identifying " a_1 " with " ba_1 " or " a_1a_1 ", and " b " with " b " or " a_1 ". That is, $G_i = \{q_0(ba_1)sq_1, q_1(ba_1)q_2, q_1(a_1a_1)q_2, q_2bsq_1, q_2a_1sq_1, q_1bvq_3, q_1a_1vq_3, q_3bfq_4, q_3a_1fq_4, q_3a_1fq_4, q_3(ba_1)fq_4, q_4b(ba_1)q_5, q_4a_1(a_1a_1)q_5, q_5(ba_1)vq_4, q_5(a_1a_1)vq_4, q_4(ba_1)(ba_1)q_6\}$. At the end of this routine, C_i is on the first square, in state q_6 , scanning " ba_1 ". Each of the first $f_i(x)+1$ squares contains the special symbol " ba_1 " or the special symbol " a_1a_1 ". All other squares contain " b " or " a_1 ".

Finally, C_i must change the number of symbols in the k -th block from $x+1$ to $f_i(x)+1$, and replace all the special symbols " ba_1 " and " a_1a_1 " with " b " and " a_1 ". This is accomplished by means of a third routine. This routine uses the special symbols on the first $f_i(x)+1$ squares as a counter and, if necessary, keeps track of the progress of the computation by means of some of the markers $\#_1, \#_2, \dots, \#_u$. When the k -th block contains exactly $f_i(x)+1$ symbols, all the markers and the special symbols " ba_1 " and " a_1a_1 " are replaced by either " b " or " a_1 ", and the machine finally goes back to the third square where it stops.

I have thus shown that any program of an $\{f_i\}$ -abacus which computes function f can be transformed into the table of a generalized Turing machine on F which computes the same function according to conventions (1) and (2). Theorem 3 is thus proved q.e.d.

We have seen (th. 2 and th. 3) that non-recursive pattern fields of a simple kind allow the computation of non-recursive functions. These results thus suggest that being a non-recursive pattern field might be a *sufficient condition* for computing non-recursive functions. This conjecture turns out to be true. I will in

fact prove below that, for any regular pattern field $F = \langle U \{f_i\} \rangle$ each numeric function f_{ei} which corresponds to f_i under an intrinsic enumeration e is COMPUTABLE on F relative to e . If, in particular, F is not recursive relative to e , at least one of the numeric functions f_{ei} is not recursive. This function, however, is COMPUTABLE on F relative to e . It thus follows that any non-recursive pattern field allows the computation of some non-recursive function.

Theorem 4 (the numeric functions which correspond to the neighbor functions of any regular pattern field are COMPUTABLE on that pattern field)

For any regular pattern field $F = \langle U \{f_i\} \rangle$, and any intrinsic enumeration e , $f_{ei}: Z^+ \rightarrow Z^+$ is COMPUTABLE on F relative to e

proof:

let us count nodes according to the intrinsic enumeration e , and let us represent input number k as a pattern inp_k which has marker $\#$ on node k and is otherwise blank.

The intrinsic enumeration e is generated by some Turing machine C which, when started in state q_0 on a completely blank pattern, reaches all the nodes. I am now going to modify C to obtain a second machine $C_\#$ which computes f_{ei} . Let A_C be the alphabet of C , where $\# \notin A_C$. The alphabet of $C_\#$ is then $A_C \cup \{\#\} = \{a_s\} = A_\#$. If G_C is the set of quadruples of C , the set of quadruples of $C_\#$ is $G_\# = G_C \cup \{q_i \# b q_{\text{jump}} q_{\text{jump}} b f_i q_{\text{mark}} q_{\text{mark}} b \# q_{\text{stop}}\}$ where q_{jump} , q_{mark} , and q_{stop} are three states which are not in the set of states $Q_C = \{q_i\}$ of C . Therefore, when $C_\#$ is started on node 0, in state q_0 , and on input pattern inp_k , $C_\#$ works exactly as C does, until it finds the marker $\#$. At that point, $C_\#$ deletes the marker, jumps to node $f_i(k)$, writes the marker $\#$ on this node, and stops in state q_{stop} .

Let $B_\# = \{a_s\} \cup \{q_r a_s\}$ be the total alphabet of $C_\#$, and define: $\pi: Z^+ \rightarrow P_F(A_\#)$, $\pi(k) =: \text{inp}_k$; $\sigma: P_F(A_\#) \rightarrow P_F(B_\#)$, $\sigma(p) =: \text{the total state of } C_\# \text{ which agrees with pattern } p \text{ on any node different from node 0 and, on node 0, has symbol } q_0 p(0)$; $c: Z^+ \rightarrow P_F(B_\#)$, $c(k) =: \sigma(\pi(k))$; $\phi: P_F(B_\#) \rightarrow P_F(A_\#)$ be the partial function which, to any total state $w \in W_\# \subset P_F(B_\#)$, assigns the pattern $p \in P_F(A_\#)$ such that $p(x) = w(x)$ if $w(x) \neq q_r a_s$, otherwise $p(x) = a_s$; $\eta: P_F(A_\#) \rightarrow Z^+$, $\eta(p) =: \text{the number of the first node which contains the marker } \#, \text{ if such a node exists, otherwise } \eta(p) =: 0$; $d: P_F(B_\#) \rightarrow Z^+$ be the partial function such that $d(w) =: \eta(\phi(w))$ if w is a total state of $C_\#$, undefined otherwise.

By the construction of $C_\#$, by the definitions of c and d , and by def. 3, $C_\#$ computes f_{ei} on F relative to c and d . Therefore, to show that f_{ei} is COMPUTABLE on F relative to e , I must only verify that (a) c is a coding of Z^+ to $P_F(B_\#)$ relative to e , and that (b) d is a decoding of $P_F(B_\#)$ to Z^+ relative to e ;

proof of (a):

suppose that $A_{\#} = \{a_s\}$ has u elements, that the blank $b = a_0$, and that the marker $\# = a_{u-1}$. The function π assigns to number k the finite pattern $\pi(k) = \text{inp}_k$. Since e is fixed, this pattern is a numeral n_e in base u which denotes the number $n = d_0 \cdot u^0 + d_1 \cdot u^1 + \dots + d_{k-1} \cdot u^{k-1} + d_k \cdot u^k$, where $d_k = u-1$ and all the other coefficients are equal to 0. Let then $\pi_e: Z^+ \rightarrow Z^+$ satisfy $\pi_e(k) =: n$. By its definition, π_e is recursive and, if $\pi(k) = n_e$, $\pi_e(k) = n$.

Suppose the state set $Q_{\#} = \{q_r\}$ of $C_{\#}$ has v elements, and that the total alphabet $B_{\#} = \{a_s\} \cup \{q_r a_s\}$ is alphabetically ordered²⁴. Then, σ assigns to the initial pattern n_e a total state $\sigma(n_e) = m_e$ which is a numeral in base $u+vu$, and which denotes the number $m = c_0 \cdot (u+vu)^0 + c_1 \cdot (u+vu)^1 + \dots + c_{k-1} \cdot (u+vu)^{k-1} + c_k \cdot (u+vu)^k$, where $c_0 = u+0u+d_0$, and all the other coefficients agree with the corresponding coefficients of the expression in base u of n . Let $\sigma_e(n) =: m$. Then, by its definition, σ_e is recursive and, if $\sigma(n_e) = m_e$, $\sigma_e(n) = m$.

Finally, let us define $c_e: Z^+ \rightarrow Z^+$, $c_e(k) = \sigma_e(\pi_e(k))$. Then, c_e is recursive and, if $c(k) = m_e$, $c_e(k) = m$. Therefore, by def. 4, C is a coding of Z^+ to $P_F(B_{\#})$ relative to e //q.e.d.//

proof of (b):

suppose the alphabet $A_{\#} = \{a_s\}$ has u elements and the state-set $Q_{\#} = \{q_r\}$ has v elements. The total alphabet $B_{\#} = \{a_s\} \cup \{q_r a_s\}$ has thus $u+vu$ elements, and let $B_{\#}$ be alphabetically ordered.

Since e is fixed, any total state $w \in W_{\#} \subset P_F(B_{\#})$ is a numeral m_e in base $u+vu$ which denotes the number $m = c_0 \cdot (u+vu)^0 + c_1 \cdot (u+vu)^1 + \dots + c_z \cdot (u+vu)^z$, where exactly one coefficient $c_x = u+ru+s$ for some r ($0 \leq r < v$) and s ($0 \leq s < u$). On the other hand, the pattern $p = \phi(m_e)$ is a numeral m_e in base u which denotes the number $n = d_0 \cdot u^0 + d_1 \cdot u^1 + \dots + d_z \cdot u^z$, where all the coefficients agree with the corresponding coefficients of m , except for $d_x = s$. Let ϕ_e satisfy $\phi_e(m) =: n$, if m is a number denoted by some total state m_e ; otherwise $\phi_e(m) =: m$. Then, ϕ_e is recursive and, if $\phi(m_e) = n_e$, $\phi_e(m) = n$.

Let $\# = a_{u-1} \in \{a_s\} = A_{\#}$, where ($0 \leq s < u$). Since e is fixed, any finite pattern p is a numeral n_e in base u which denotes the number n and, by the definition of η , η assigns to pattern n_e the index y of the smallest term $d_y u^y$ in the u -ary expression of n such that $d_y = u-1$, if such a term exists; otherwise, $\eta(n_e) = 0$. Let then $\eta_e: Z^+ \rightarrow Z^+$ satisfy: $\eta_e(n) =$ the index y of the smallest term $d_y u^y$ in the u -ary expression of n such that $d_y = u-1$, if such a term exists; $\eta_e(n) = 0$, otherwise. By its definition, η_e is recursive and, if $\eta(n_e) = h$, $\eta_e(n) = h$.

Finally, let us define $d_e: Z^+ \rightarrow Z^+$, $d_e(m) = \eta_e(\phi_e(m))$. Then, d_e is recursive and, if $d(m_e) = h$, $d_e(m) = h$. Therefore, by def. 5, d is a decoding of $P_F(B_{\#})$ to Z^+ //q.e.d.//

Theorem 4 is thus proved q.e.d.

²⁴ Therefore, according to this order, the symbol a_s corresponds to number s , and the pair $q_r a_s$ corresponds to number $u+ru+s$.

An immediate consequence of theorem 4 is that any regular, but not recursive, pattern field allows the computation of non-recursive functions:

Corollary 4.1 (being a regular and non-recursive pattern field is sufficient for COMPUTING non-recursive functions)

For any regular pattern field F , and any intrinsic enumeration e , if F is not recursive relative to e , there is a non-recursive function which is COMPUTABLE on F relative to e

proof:

since F is not recursive relative to e , at least one of the numeric functions $\{f_{e_i}\}$ which correspond to the neighbor functions $\{f_i\}$ is not recursive. By th. 4, this non-recursive function is COMPUTABLE on F relative to e q.e.d.

If we think of a regular pattern field $F = \langle U \{f_i\} \rangle$ as an infinite world in which generalized Turing machines 'live and operate', then the set of the neighbor functions $\{f_i\}$ can be thought as specifying the 'spatial structure' of this world. Then, according to this interpretation, theorem 4 tells us that any numeric representation of this structure can always be computed in that world, even if this representation turns out not to be recursive.

4. Is a non-recursive pattern field necessary for computing non-recursive functions?

We have seen in the previous section that any non-recursive function can be computed on an appropriate pattern field (see th. 2). However, the neighborhood structure of the pattern field which permits this computation turns out to be non-recursive. This is in fact a special case of a more general result. I will prove below (th. 5) that *any numeric function COMPUTABLE on an a regular pattern field $F = \langle U \{f_j\} \rangle$ relative to an intrinsic enumeration e is (partial) recursive relative to the functions $\{f_{e_j}\}$ which correspond to the neighbor functions.* In particular, if F is recursive, the class of the (partial) recursive functions relative to $\{f_{e_j}\}$ is identical to the class of the (partial) recursive functions. It thus follows that any numeric function COMPUTABLE on a recursive pattern field relative to an intrinsic enumeration e is (partial) recursive. In other words, being a non-recursive pattern field is a necessary condition for the computability of non-recursive functions.

The proof that all the numeric functions COMPUTABLE on a regular pattern field $F = \langle U \{f_j\} \rangle$ are (partial) recursive relative to $\{f_{e_j}\}$ (see th. 5 below) is a natural generalization of the analogous proof for ordinary Turing machines and (partial) recursive functions. To understand the strategy which I will use to prove theorem 5, it is thus useful to review first the main ideas of the corresponding proof for ordinary Turing machines.

The computation of a numeric function always involves three different steps. Suppose we want to use a Turing machine C to compute the numeric function $f(n)$. First, we need to represent, or *code*, the number n as a total state of the Turing

machine C . Second, once the initial total state which codes the number n is fixed, *we let the machine start on that state, and we then wait until it stops*. Third, given the final total state of C , we need to *decode* the number represented by that total state. If, for any n , this number is equal to $f(n)$, we conclude that f is computable by C .

Each of these three steps can be identified with a function. The first step consists in applying a coding function c to the number n . The second step corresponds to a (partial) function ρ_C which, to the initial total state $c(n)$, associates the final total state of the computation, if the Turing machine C stops. Otherwise, $\rho_C(c(n))$ is undefined. Finally, the third step applies a decoding function d which retrieves the number represented by the final total state $\rho_C(c(n))$. Therefore, f is computable by C just in case, for any n , $f(n) = d(\rho_C(c(n)))$, if $f(n)$ is defined; otherwise, $\rho_C(c(n))$ is undefined.

Suppose now that we have a way of representing total states as numbers. Then, under this representation, the functions d , ρ_C , and c correspond to three numeric functions d_e , ρ_{eC} , and c_e . It thus follows that, for any n , $d(\rho_C(c(n))) = d_e(\rho_{eC}(c_e(n))) = f(n)$. Now, if we can prove that each of the numeric functions d_e , ρ_{eC} , and c_e is (partial) recursive, then f must be (partial) recursive as well. If the Turing machine C which computes f is an ordinary one, a natural representation of total states as numbers is the one induced by the standard

enumeration e of the tape²⁵. Furthermore, if the computation is performed with the usual conventions, we have already proved that the functions c_e and d_e (which correspond to c and d under such a representation) are recursive²⁶. Therefore, we need only prove that the function p_{eC} is (partial) recursive. In the first place, it is quite obvious that p_{eC} can be recursively defined if the numeric function $g[G_C]_e$ which corresponds to the total transition function $g[G_C]$ of C is recursive. And this, in turn, can be proved by showing that the numeric functions which correspond to each quadruple in G_C are recursive, and by then defining $g[G_C]_e$ by cases. It is important to notice that this proof depends on the fact that the numeric functions which correspond to the neighbor functions Left and Right are recursive. In other words, the numeric function which corresponds to the total transition function of an ordinary Turing machine is recursive *because* the tape is a recursive pattern field. We will see shortly that this is always true: if a pattern field F is recursive relative to an intrinsic enumeration e , and $g[G_C]_e$ is the function which corresponds to the total transition function of a generalized Turing machine on F , then $g[G_C]_e$ is recursive.

The proof for the general case, when C is a Turing machine on an arbitrary

²⁵ The standard enumeration of the tape is defined in example 1.2.1. Recall that, when the order of the squares is fixed according to the standard enumeration e , each total state of a Turing machine C is a numeral m_e in base $k+r$, where k is the number of symbols in the alphabet $A_C = \{a_i\}$ and r is the number of internal states in $Q_C = \{q_j\}$. Also, if the total alphabet $B_C = \{a_i\} \cup \{q_j a_i\}$ is alphabetically ordered, number m is denoted by some total state just in case the expression of m in base $k+r$ has exactly one coefficient equal to $k+jk+i$, for some j ($0 \leq j < r$) and i ($0 \leq i < k$). This condition is recursive. Therefore, the standard enumeration of the tape induces a bijection between a recursive subset of the non-negative integers and the set of all total states.

²⁶ See examples 4.1 and 5.1.

regular pattern field $F = \langle U f_i \rangle$ with a fixed intrinsic enumeration e , follows an analogous strategy. I will first prove two lemmas. The first affirms that, if the numeric function $g[G_C]_e$ is recursive relative to $\{f_{ei}\}$, then the numeric function ρ_{eC} is (partial) recursive relative to $\{f_{ei}\}$. The second states that $g[G_C]_e$ is recursive relative to $\{f_{ei}\}$. These two lemmas, together with the definition of COMPUTABLE function (def. 6), thus imply: for any regular pattern field $F = \langle U \{f_i\} \rangle$, and any intrinsic enumeration e , if a numeric function f is COMPUTABLE on F relative to e , then f is (partial) recursive relative to $\{f_{ei}\}$. I now give the details of the proof.

Let C be a Turing machine on a regular pattern field $F = \langle U \{f_i\} \rangle$ with a fixed intrinsic enumeration e . Let $A_C = \{a_l\}$ ($0 \leq l < k$) be the alphabet of C and $Q = \{q_j\}$ ($0 \leq j < r$) be the state-set of C . Let the elements of the total alphabet $B_C = \{a_l\} \cup \{q_j a_l\}$ be alphabetically ordered²⁷. Since e is fixed, any total state n_e is a numeral in base $k+r$ which denotes the number n . Let $g[G_C]$ be the total transition function of the Turing machine C , and let $g[G_C]_e: Z^+ \rightarrow Z^+$ satisfy $g[G_C]_e(n) = k$, if $g[G_C](n_e) = k_e$. If the number n is not denoted by any total state, I set $g[G_C]_e(n) = n$. Let ρ_C be the function which, to any total state n_e , associates the final total state r_e of the computation which starts with n_e . If such a computation does not end, ρ_C is undefined. Let $\rho_{eC}: Z^+ \rightarrow Z^+$ satisfy $\rho_{eC}(n) = r$, if $\rho_C(n_e) = r_e$. If $\rho_C(n_e)$ is undefined, $\rho_{eC}(n)$ is undefined and, if the number n is not denoted by any total state, I set $\rho_{eC}(n) = n$. Let $\{f_{ei}\}$ be the numeric functions which

²⁷ Therefore, according to this order, the symbol a_l corresponds to number l , and the pair $q_j a_l$ corresponds to number $k+jk+l$.

correspond to the neighbor functions $\{f_i\}$ of F . I show below that ρ_{eC} is (partial) recursive relative to $\{f_{ei}\}$ if $g[G_C]_e$ is recursive relative to $\{f_{ei}\}$.

Lemma 5.1

If $g[G_C]_e$ is recursive relative to $\{f_{ei}\}$, then ρ_{eC} is (partial) recursive relative to $\{f_{ei}\}$

proof:

suppose that $g[G_C]_e$ is recursive relative to $\{f_{ei}\}$. In the first place, I recursively define the m -th iteration of $g[G_C]_e$ as follows: $\psi_e(0, n) = n$, $\psi_e(m+1, n) = g[G_C]_e(\psi_e(m, n))$. I then define a function H_e which checks whether the computation which starts with total state n_e halts at time m . This function returns 0 if the computation halts at time m , 1 otherwise. Recall that a computation halts just in case the total states at two consecutive times are identical. H_e can thus be recursively defined as follows: $H_e(m, n) = 0$, if $\psi_e(m+1, n) = \psi_e(m, n)$; 1, otherwise. I now define a function T_e which returns the time at which the computation halts. If the computation does not halt, T_e is undefined. Obviously, T_e can be obtained from H_e by applying the minimization operator, that is: $T_e(n) =$ the least m such that $H_e(m, n) = 0$. T_e is thus (partial) recursive relative to $\{f_{ei}\}$. Finally, the function ρ_{eC} can be expressed in terms of ψ_e and T_e . In fact, $\rho_{eC}(n) = \psi_e(T_e(n), n)$. Therefore, ρ_{eC} is (partial) recursive relative to $\{f_{ei}\}$ q.e.d.

I now prove that the function which corresponds to the transition function of a generalized Turing machine on a regular pattern field $F = \langle U \{f_i\} \rangle$ is recursive relative to $\{f_{ei}\}$:

Lemma 5.2 (for any regular pattern field, the numeric function which corresponds to the transition function of an arbitrary Turing machine is recursive relative to the numeric functions which correspond to the neighbor functions)

For any regular pattern field $F = \langle U \{f_i\} \rangle$, and any intrinsic enumeration e , $g[G_C]_e$ is recursive relative to $\{f_{ei}\}$

proof:

let $A_C = \{a_i\}$ ($0 \leq i < k$) be the alphabet of Turing machine C , and $Q_C = \{q_j\}$ ($0 \leq j < r$) be the state-set of C . Let the elements of the total alphabet $B_C = \{a_i\} \cup \{q_j a_i\}$ be

alphabetically ordered²⁸. The quadruples of C are of two kinds: $q_j a_l a_z q_t$ and $q_j a_l f_i q_t$. I first show that the numeric function which corresponds to each type of quadruple is partial recursive relative to $\{f_{e_i}\}$, and that its domain is a recursive subset of Z^+ ; $g[G]_e$ can then be defined by cases by means of these functions, and it is thus recursive relative to $\{f_{e_i}\}$.

Since any total state m_e is a numeral in base $k+rk$, the decomposition of the number m denoted by m_e is:

$$m = c_0 \cdot (k+rk)^0 + c_1 \cdot (k+rk)^1 + \dots + c_u \cdot (k+rk)^u$$

where exactly one coefficient $c_s = k+jk+l$, for some j ($0 \leq j < r$) and l ($0 \leq l < k$).

Let $[q_j a_l a_z q_t]_e$ be the numeric function which corresponds to a quadruple of the first type. This function thus transforms the number m into the number n such that

$$n = d_0 \cdot (k+rk)^0 + d_1 \cdot (k+rk)^1 + \dots + d_u \cdot (k+rk)^u$$

where all the coefficients agree with the corresponding coefficients of m , except for $d_s = k+tk+z$. It thus follows that $[q_j a_l a_z q_t]_e$ is partial recursive, and its domain is a recursive subset of Z^+ . In fact, $[q_j a_l a_z q_t]_e(m)$ is defined just in case there is exactly one coefficient c_s such that $c_s = k+jk+l$.

Let $f = f_{e_i}$ be the numeric function which corresponds to the neighbor function f_i under the intrinsic enumeration e , and $[q_j a_l f_i q_t]_e$ be the numeric function which corresponds to a quadruple of the second type. This function thus transforms the number m into the number n such that

$$n = d_0 \cdot (k+rk)^0 + d_1 \cdot (k+rk)^1 + \dots + d_v \cdot (k+rk)^v$$

where, if $f(s) \neq s$, all the coefficients agree with the corresponding coefficients of m , except for $d_s = l$, and for $d_{f(s)} = k+tk+c_{f(s)}$. If $f(s) = s$, all the coefficients agree with the corresponding coefficients of m , except for $d_s = k+tk+l$. It thus follows that $[q_j a_l f_i q_t]_e$ is partial recursive relative to $\{f_{e_i}\}$, and its domain is a recursive subset of Z^+ . In fact, $[q_j a_l f_i q_t]_e(m)$ is defined just in case there is exactly one coefficient c_s such that $c_s = k+jk+l$.

Finally, since the domains of the partial numeric functions which correspond to each quadruple are recursive and mutually exclusive, $g[G_C]_e$ can be recursively defined by cases. Therefore, $g[G_C]_e$ is recursive relative to $\{f_{e_i}\}$ q.e.d.

The previous two lemmas, together with the definition of COMPUTABLE function, imply that any numeric function which is COMPUTABLE on a regular

²⁸ Therefore, according to this order, the symbol a_l corresponds to number l , and the pair $q_j a_l$ corresponds to number $k+jk+l$.

pattern field is (partial) recursive relative to the numeric functions which correspond to the neighbor functions:

Theorem 5 (any numeric function COMPUTABLE on a regular pattern field is (partial) recursive relative to the numeric functions which correspond to the neighbor functions)

For any regular pattern field $F = \langle U \{f_i\} \rangle$, and any intrinsic enumeration e , if $g: Z^+ \rightarrow Z^+$ is COMPUTABLE on F relative to e , then g is (partial) recursive relative to $\{f_{e_i}\}$

proof:

suppose F is regular, and g is COMPUTABLE on F relative to e . Then, by definition 6, there is C , c , and d such that C is a generalized Turing machine on F , c is a coding of Z^+ to $P_F(B_C)$ relative to e , d is a decoding of $P_F(B_C)$ relative to e , and C computes g on F relative to c and d . By definition 3, C computes g on F relative to c and d just in case, for any n , $d(\rho_C(c(n))) = g(n)$, if $g(n)$ is defined; otherwise, $\rho_C(c(n))$ is undefined. Therefore, since C computes g on F relative to c and d , $d_e(\rho_{eC}(c_e(n))) = g(n)$ if $g(n)$ is defined; otherwise, $\rho_{eC}(c_e(n))$ is undefined. By definitions 4 and 5, the numeric functions c_e and d_e are recursive. By lemmas 5.2 and 5.1, since F is regular, ρ_{eC} is (partial) recursive relative to $\{f_{e_i}\}$. It thus follows that g is (partial) recursive relative to $\{f_{e_i}\}$ q.e.d.

I finally deduce six corollaries which sum up the results of this section and of the previous one.

Corollary 5.1 (any numeric function COMPUTABLE on a recursive pattern field is (partial) recursive)

If a pattern field $F = \langle U \{f_i\} \rangle$ is recursive relative to an intrinsic enumeration e , and $g: Z^+ \rightarrow Z^+$ is COMPUTABLE on F relative to e , then g is (partial) recursive

proof:

by theorem 5, g is (partial) recursive relative to $\{f_{e_i}\}$. Since F is recursive relative to e , each function f_{e_i} is recursive. Therefore, g is (partial) recursive q.e.d.

Corollary 5.2 (the class of the (partial) recursive functions is identical to the class of the functions COMPUTABLE on recursive pattern fields)

$g: Z^+ \rightarrow Z^+$ is (partial) recursive iff there is a pattern field F and an intrinsic enumeration e such that F is recursive relative to e , and g is COMPUTABLE on F relative to e

proof:

right-left follows from corollary 5.1. The implication from left to right follows from theorem 1 q.e.d.

Corollary 5.3 (being a regular and non-recursive pattern field is a necessary and sufficient condition for COMPUTING non-recursive functions)

there is a non-recursive function $g: Z^+ \rightarrow Z^+$ such that g is COMPUTABLE on F relative to e iff F is regular and F is not recursive relative to e

proof:

right-left is corollary 4.1. Left-right follows from definition 6 and corollary 5.1 q.e.d.

Corollary 5.4 (the class of the (partial) recursive functions relative to $\{f_i\}$ is identical to the class of the functions COMPUTABLE on²⁹ $\langle Z^+ \{s v\} \cup \{f_i\} \rangle$ relative to the identity function on Z^+)

$g: Z^+ \rightarrow Z^+$ is (partial) recursive relative to $\{f_i\}$ iff g is COMPUTABLE on $\langle Z^+ \{s v\} \cup \{f_i\} \rangle$ relative to the identity function e on Z^+

proof:

left-right is theorem 3. Right-left follows from theorem 5 q.e.d.

Corollary 5.5 (the class of the (partial) functions computable by oracle machines is identical to the class of the functions COMPUTABLE on $\langle Z^+ \{s v\} \cup \{f_X\} \rangle$ relative to the identity function on Z^+ , where f_X is the characteristic function of $X \subseteq Z^+$)

A (partial) function $g: Z^+ \rightarrow Z^+$ is computable by an oracle machine whose oracle computes the characteristic function f_X of $X \subseteq Z^+$ iff g is COMPUTABLE on $\langle Z^+ \{s v\} \cup \{f_X\} \rangle$ relative to the identity function e on Z^+

²⁹ Recall that s is the successor function, and v the predecessor function (where $v(0) =: 0$).

proof:

oracle machines whose oracles compute f_x compute exactly the class of the (partial) recursive functions relative to $\{f_x\}$. Therefore, the thesis follows from corollary 5.4 q.e.d.

Corollary 5.6 (the class of the (partial) recursive functions is identical to the class of the functions COMPUTABLE on $\langle Z^+ \{s v\} \rangle$ relative to the identity function on Z^+)

$g: Z^+ \rightarrow Z^+$ is (partial) recursive iff g is COMPUTABLE on $\langle Z^+ \{s v\} \rangle$ relative to the identity function e on Z^+

proof:

$g: Z^+ \rightarrow Z^+$ is (partial) recursive iff g is (partial) recursive relative to the empty set. Therefore, from corollary 5.4, the thesis follows q.e.d.

5. Generalized computational systems on regular pattern fields

In chapter 1, I have proposed to identify a computational system with a discrete dynamical system (a cascade) which can be *effectively described*. This means that the phase space of an *isomorphic* cascade is *decidable*, and that its transition function is an *effective transformation of finite symbol structures*. If we assume that any effective transformation of finite symbol structures can be reduced to a function computable by an ordinary Turing machine, we can then define a computational system as a cascade isomorphic to a second cascade whose phase space is decidable, and whose transition function is Turing computable (ch. 1, def. 3). However, the results of this chapter show that *the concept of an effective transformation of finite symbol structures is not absolute, but instead depends on the relational structure of the infinite support on which the elementary symbols are written*. In fact, if a generalized Turing machine operates on a regular but non-recursive pattern field, its transformations of finite symbol structures may permit the computation of non-recursive functions. On the other hand, Turing machines which operate on recursive pattern fields can only compute recursive functions. The goal of this section is to give a more general definition of a computational system which takes into account these results.

In the first place, *the concept of an effective transformation of finite symbol structures should be relativized to the pattern field on which these structures are written and manipulated*. In other words, I propose the following generalization of Turing's thesis. Let F be a given pattern field, and let X and Y be two subsets of

the set $P_F(A)$ of all finite patterns of F relative to some alphabet A . Then, a (partial) function $f: X \rightarrow Y$ is effective, just in case f is COMPUTABLE on F relative to some intrinsic enumeration³⁰ of F . In the second place, *also the concept of a computational system should be relativized to the pattern field which allows us to describe the system in an effective way*. I will thus define below the concept of a *computational system on pattern field F* . This concept is in fact a generalization of the one I have defined in chapter 1, for it reduces to the previous one when the pattern field F is identified with the tape of an ordinary Turing machine³¹.

Before giving this new definition, however, I need a few preliminaries. In the first place, I need to modify the definition of a COMPUTABLE function (def. 6) in order to include the case of a *symbolic function*, that is a function $f: X \rightarrow Y$ such that X and Y are subsets of the set $P_F(A)$ of all *finite patterns* of F relative to alphabet A . This can easily be obtained by introducing first the two concepts of a coding of X to $P_F(B_C)$ relative to an intrinsic enumeration e , and of a decoding of $P_F(B_C)$ to Y relative to e ³². These two definitions are analogous to definitions 4 and 5. That is: c is a coding of X to $P_F(B_C)$ relative to e iff: c is a coding of X to $P_F(B_C)$, and there is a recursive function c_e such that, if $c(k_e) = n_e$, then $c_e(k) = n$;

³⁰ See below for the definition of " f is a COMPUTABLE function on F relative to an intrinsic enumeration e " when f is a *symbolic function*, that is a function whose domain and codomain are included in $P_F(A)$, for some alphabet A .

³¹ More precisely, it is possible to prove the following: S is a computational system (ch. 1, def. 3) iff S is a generalized computational system on F (def. 8 below), where F is the tape of an ordinary Turing machine.

³² Recall that $P_F(B_C)$ is the set of all finite patterns of F relative to the total alphabet B_C of Turing machine C .

d is a decoding of $P_F(B_C)$ to Y relative to e iff: d is a decoding of $P_F(B_C)$ to Y , and there is a recursive function d_e such that, if $d(n_e) = k_e$, then $d_e(n) = k$. I then modify definition 6 as follows: f is a COMPUTABLE function on F relative to e iff: f is a (partial) function from X to Y , where X and $Y \subseteq P_F(A)$ for some alphabet A , there is a generalized Turing machine C on F , and there are c and d such that c is a coding of X to $P_F(B_C)$ relative to e , d is a decoding of $P_F(B_C)$ to Y relative to e , and C computes f on F relative to c and d .

In the second place, I need to define the concept of *characteristic function* for a subset X of $P_F(A)$. That is: f_X is a characteristic function of X in $P_F(A)$ iff: $X \subseteq P_F(A)$, there is $\{p_0, p_1\} \subseteq P_F(A)$ such that $f_X: P_F(A) \rightarrow \{p_0, p_1\}$, and $f(x) = p_1$ if $x \in X$, $f(x) = p_0$ if $x \notin X$. Finally, I can define:

Definition 8 (generalized computational systems on a regular pattern field)

S is a generalized computational system on F iff:

S is a cascade, F is a regular pattern field, and there is a second cascade $S_1 = \langle T, M_1, H \rangle$ such that

- (1) S is isomorphic to S_1 ;
- (2) there is a finite alphabet A such that $M_1 \subseteq P_F(A)$ and, for any intrinsic enumeration e of F ,
 - (a) some characteristic function of M_1 in $P_F(A)$ is COMPUTABLE on F relative to e ;
 - (b) the transition function $H: M_1 \rightarrow M_1$ is COMPUTABLE on F relative to e and, if $T = Z$, also H^{-1} (the inverse of H) is COMPUTABLE on F relative to e .

Intuitively, if F is a regular pattern field, all generalized Turing machines on F are computational systems on F . Therefore, if the previous definition is materially correct, we should be able to prove that it is satisfied by any Turing machine which

operates on a fixed regular pattern field. The following theorem shows that this is in fact true.

Theorem 6 (all generalized Turing machines on a regular pattern field F are generalized computational systems on that pattern field)

For any regular pattern field F , if $C_1 = \langle Z^+ M_1 g[G_1] \rangle$ is a generalized Turing machine on F , then C_1 is a generalized computational system on F

proof:

let C_1 be a generalized Turing machine on pattern field $F = \langle U \{f_i\} \rangle$. Let $A_1 = \{a_i\}$ ($0 \leq i < k$) be the alphabet of C_1 , and $Q_1 = \{q_j\}$ ($0 \leq j < r$) be the state-set of C_1 . I now construct a second Turing machine C_2 which computes the transition function $g[G_1]$ of C_1 on the same pattern field F . Let $B_1 = \{a_i\} \cup \{q_j a_i\} = A_2$ be the alphabet of C_2 . The quadruples of C_1 are of two types: $q_j a_i a_s q_u$ and $q_j a_i f_i q_u$. For each quadruple of the first type, C_2 has a subroutine specified by the quadruple $\mathbf{q}_{start}(q_j a_i)(q_u a_s) \mathbf{q}_{stop}$. For each quadruple of the second type, C_2 has a subroutine specified by the quadruple $\mathbf{q}_{start}(q_j a_i) a_i \mathbf{q}_{move(u)}, \mathbf{q}_{move(u)} a_i f_i q_u, \mathbf{q}_u a_0(q_u a_0) \mathbf{q}_{stop}, \dots, \mathbf{q}_u a_{k-1}(q_u a_{k-1}) \mathbf{q}_{stop}$. Suppose that C_2 is started on a total state of C_1 , in internal state \mathbf{q}_{start} , on the node which contains the pair $q_j a_i$. If this pair corresponds to a quadruple of the first type, C_2 replaces $q_j a_i$ by $q_u a_i$ and then stops in state \mathbf{q}_{stop} . Otherwise, it first replaces $q_j a_i$ by a_i , it then moves to a new node according to f_i , adds the symbol q_u to the symbol on this node, and stops in state \mathbf{q}_{stop} .

Let B_2 be the total alphabet of C_2 , and $M_1 \subset P_F(A_2) = P_F(B_1)$ be the set of all total states of C_1 . Define: $c: M_1 \rightarrow P_F(B_2)$, $c(w) =$: the total state of C_2 which agrees with w except for the node which contains a pair $q_j a_i$; $c(w)$ assigns to this node $\mathbf{q}_{start}(q_j a_i)$. Let $S_2 \subset P_F(B_2)$ be the set of all total states v such that C_2 stops on v when started on $c(w)$, for some $w \in M_1$. Let $d: P_F(B_2) \rightarrow M_1$ be the partial function which, to any $s \in S_2$, assigns the total state of C_1 obtained from s by replacing $\mathbf{q}_{stop}(q_u a_s)$ with $q_u a_s$; if $s \notin S_2$, d is undefined. Then by the definitions of c and d , and by def. 3, C_2 computes the transition function $g[G_1]$ of C_1 on F relative to c and d . Therefore, to show that $g[G_1]$ is COMPUTABLE on F relative to any intrinsic enumeration e , I must only verify that (i) c is a coding of M_1 to $P_F(B_2)$ relative to e , and that (ii) d is a decoding of $P_F(B_2)$ to M_1 relative to e .

Let e be an arbitrary intrinsic enumeration of F , and let $A_2 = \{a_i\} \cup \{q_j a_i\}$ be alphabetically ordered. Since e is fixed, any pattern $p \in P_F(A_2)$ is a numeral m_e in base $k+rk$ which denotes the number $m = c_0 \cdot (k+rk)^0 + c_1 \cdot (k+rk)^1 + \dots + c_z \cdot (k+rk)^z$. Furthermore, m_e is a total state of C_1 just in case exactly one coefficient c_x is equal to $k+jk+i$, for some j ($0 \leq j < r$) and i ($0 \leq i < k$). Let t be the number of elements of the total alphabet B_2 of C_2 . Then, the function c assigns to m_e a numeral n_e in base t which denotes the number $n = d_0 \cdot t^0 + d_1 \cdot t^1 + \dots + d_z \cdot t^z$, where all the coefficients agree with the corresponding coefficients of m except for $d_x = (k+rk)+0(k+rk)+c_x$ (assuming that $\mathbf{q}_{start} = \mathbf{q}_0$, and that B_2 is alphabetically ordered). Define $c_e: Z^+ \rightarrow Z^+$, $c_e(m) =: n$ if m_e is a total state of C_1 , $c_e(m) =: m$ otherwise. Then, c_e is recursive and,

if $c(m_e) = n_e$, $c_e(n) = m$. Therefore, c is a coding of M_1 to $P_F(B_2)$ relative to e .

Let e be an arbitrary intrinsic enumeration of F , let $Q_2 = \{q_j\}$ be the state-set of C_2 , where $0 \leq j < h$, and assume that $q_{stop} = q_{h-1}$. Also assume that the total alphabet B_2 of C_2 has t elements, where $t = (k+rk)+h(k+rk)$, and that B_2 is alphabetically ordered. Since e is fixed, any pattern $p \in P_F(B_2)$ is a numeral n_e in base t which denotes the number $n = d_0 \cdot t^0 + d_1 \cdot t^1 + \dots + d_z \cdot t^z$. Furthermore, if $n_e \in S_2$, exactly one coefficient $d_x = (k+rk)+(\mathbf{h}-1)(k+rk)+(k+uk+l)$, for some u ($0 \leq u < r$) and l ($0 \leq l < k$). The function d assigns to n_e a numeral m_e in base $k+rk$ which denotes the number $m = c_0 \cdot (k+rk)^0 + c_1 \cdot (k+rk)^1 + \dots + c_z \cdot (k+rk)^z$, where all the coefficients agree with the corresponding coefficients of n except for $c_x = k+uk+l$. Define $d_e: Z^+ \rightarrow Z^+$, $d_e(n) =: m$ if exactly one coefficient d_x in the expression in base t of n_e is equal to $(k+rk)+(\mathbf{h}-1)(k+rk)+(k+uk+l)$, $d_e(n) =: n$ otherwise. Then, d_e is recursive and, if $d(n_e) = m_e$, $d_e(n) = m$. Therefore, d is a decoding of $P_F(B_2)$ to M_1 relative to e .

I now prove that, for any intrinsic enumeration e of F , a characteristic function g_1 of M_1 in $P_F(B_1)$ is COMPUTABLE on F relative to e . Let us count nodes according to the intrinsic enumeration e , and let $g_1: P_F(B_1) \rightarrow \{p_0, p_1\} \subseteq P_F(B_1)$, where p_0 is the completely blank pattern and p_1 is the pattern which has the pair $q_0 a_0$ on the first node, and is otherwise blank. Let $g_1(p) =: p_1$ if $p \in M_1$, $g_1(p) =: p_0$ if $p \notin M_1$. The intrinsic enumeration e is generated by some Turing machine C_e which, when started in state q_0 , on node x of a completely blank pattern, reaches all the nodes. I am now going to modify C_e to obtain another machine $C_\#$ which computes g_1 . Let $A_e = \{a_u\}$ ($0 \leq u < n$) be the alphabet of C_e , and $Q_e = \{q_v\}$ ($0 \leq v < m$) be the state-set of C_e . If the total alphabet B_1 of C_1 is $\{a_i\} \cup \{q_j a_i\}$, the alphabet of $C_\#$ is $A_\# = \{a_i a_u\} \cup \{q_j a_i a_u\} \cup \{\#\}$. The quadruples of C_e are of two types $q_v a_u a_s q_t$ and $q_v a_u f_i q_t$. For each quadruple of the first type, $C_\#$ has a subroutine specified by the quadruples: $\{q_v(a_i a_u)(a_i a_s)q_t\} \cup \{q_v(q_j a_i a_u)(a_i a_s)q_{t\#}\} \cup \{q_{v\#}(a_i a_u)(a_i a_s)q_{t\#}\}$. For each quadruple of the second type, $C_\#$ has a subroutine specified by the quadruples: $\{q_v(a_i a_u) f_i q_t\} \cup \{q_v(q_j a_i a_u)(a_i a_u)q_{v\#}\} \cup \{q_{v\#}(a_i a_u) f_i q_{t\#}\} \cup \{q_{v\#}(a_i a_u) f_i q_{t\#}\}$. Finally, $C_\#$ has the halting conditions: $\{q_{v\#\#}q_{no}, q_{v\#\#}q_{yes}\} \cup \{q_{v\#}(q_j a_i a_u)(q_j a_i a_u)q_{no}\}$. Suppose that $C_\#$ is started in state q_0 , on the first node x (according to the intrinsic enumeration e) and on a pattern $p_\#$ which represents pattern $p \in P_F(B_1)$ in the following way. The marker $\#$ is located on the first node after the last non-blank node of p . All nodes after the node where $\#$ is located are blank (where the blank is symbol $a_0 a_0$). For any other node y , if $p(y) = a_i$, $p_\#(y) = a_i a_0$; if $p(y) = q_j a_i$, $p_\#(y) = q_j a_i a_0$. If p is completely blank, the marker $\#$ is located on the first node x . There are three possible cases: (1) there is no node y such that $p(y) = q_j a_i$; (2) there is exactly one node y such that $p(y) = q_j a_i$; (3) there are at least two node y and z such that $y = q_j a_i$ and $z = q_t a_s$. Furthermore, $p \in M_1$ just in case (2) is satisfied. $C_\#$ is designed so that, when started in state q_0 on the first node x of pattern $p_\#$, it stops in state q_{yes} if (2) holds; if (1) or (3) holds, instead, $C_\#$ stops in state q_{no} . In fact, $C_\#$ moves from node to node looking for either the marker $\#$ or a triple $q_j a_i a_u$. If it finds the marker $\#$ first, it stops in state q_{no} (case 1). If it finds a triple $q_j a_i a_u$ first, it memorizes this fact by going into state $q_{t\#}$, and then starts looking again for the marker or a second triple. If it finds the marker $\#$ first, it stops in state q_{yes} (case 2). Otherwise, it stops in state q_{no} (case 3).

Let $B_\#$ be the total alphabet of $C_\#$ and define $c: P_F(B_1) \rightarrow P_F(B_\#)$, $c(p) =:$ the pattern which has $q_0 p(x) a_0$ on the first node x and agrees with $p_\#$ on all other nodes, if p is not completely blank; otherwise $c(p)$ has $q_0 \#$ on the first node x and all other nodes are

blank (where the blank is symbol $a_0 a_0$); $d: P_F(B_\#) \rightarrow \{p_0, p_1\}$ be the partial function which assigns p_0 to any pattern which contains exactly one node with either $q_{no}\#$ or $q_{no}(q_j a_i a_u)$, and assigns p_1 to any pattern which contains exactly one node with $q_{yes}\#$. Then by construction of $C_\#$, by the definitions of c and d , and by def. 3, $C_\#$ computes g_1 on F relative to c and d . Therefore, to show that g_1 is COMPUTABLE on F relative to e , I must only verify that (i) c is a coding of $P_F(B_1)$ to $P_F(B_\#)$ relative to e , and that (ii) d is a decoding of $P_F(B_\#)$ to $\{p_0, p_1\}$ relative to e .

Since e is fixed, and B_1 has $k+rk$ elements, any pattern $p \in P_F(B_1)$ is a numeral m_e in base $k+rk$ which denotes the number $m = c_0 \cdot (k+rk)^0 + c_1 \cdot (k+rk)^1 + \dots + c_z \cdot (k+rk)^z$. Since the alphabet A_e of C_e has n elements, the alphabet $A_\# = \{a_i a_u\} \cup \{q_j a_i a_u\} \cup \{\#\}$ of $C_\#$ has $kn+rkn+1$ elements. Let the marker $\#$ correspond to number $kn+rkn$, and $A_\#$ be alphabetically ordered. Therefore, $a_i a_u$ corresponds to number $in+u$ and $q_j a_i a_u$ corresponds to number $kn+jkn+in+u$. Suppose the state set $Q_\#$ has t elements. Then the total alphabet $B_\#$ of $C_\#$ has $(kn+rkn+1)+t(kn+rkn+1)$ elements. Suppose $B_\#$ is alphabetically ordered. Then, $q_0(a_i a_u)$ corresponds to number $(kn+rkn+1)+0(kn+rkn+1)+(in+u)$, $q_0(q_j a_i a_u)$ corresponds to number $(kn+rkn+1)+0(kn+rkn+1)+(kn+jkn+in+u)$, and $q_0\#$ corresponds to number $(kn+rkn+1)+0(kn+rkn+1)+(kn+rkn)$. The function c assigns to pattern m_e a total state of $C_\#$ which is a numeral v_e in base $(kn+rkn+1)+t(kn+rkn+1) = h$ and denotes the number $v = d_0 \cdot h^0 + d_1 \cdot h^1 + \dots + d_z \cdot h^z + d_{z+1} \cdot h^{z+1}$ where, if $m \neq 0$, $d_{z+1} = kn+rkn$, $d_0 = (kn+rkn+1)+(in+0)$ if $c_0 = i$, $d_0 = (kn+rkn+1)+(kn+jkn+in+0)$ if $c_0 = k+jk+i$, any other coefficient $d_x = in+0$ if $c_x = i$, $d_x = kn+jkn+in+0$ if $c_x = k+jk+i$; if $m = 0$, then $v = d_0 h^0$, where $d_0 = (kn+rkn+1)+(kn+rkn)$. Define $c_e: Z_+ \rightarrow Z_+$, $c_e(m) = v$. Then, c_e is recursive and, if $c(m_e) = v_e$, $c_e(m) = v$. Therefore, c is a coding of $P_F(B_1)$ to $P_F(B_\#)$ relative to e .

Suppose that q_{yes} corresponds to number $t-2$, and that q_{no} corresponds to number $t-1$. Then, $q_{yes}\#$ corresponds to number $Y = (kn+rkn+1)+(t-2)(kn+rkn+1)+(kn+rkn)$, $q_{yes}\#$ corresponds to number $N = (kn+rkn+1)+(t-1)(kn+rkn+1)+(kn+rkn)$, and $q_{no}(q_j a_i a_u)$ corresponds to number $N_{jii} = (kn+rkn+1)+(t-1)(kn+rkn+1)+(kn+jkn+in+u)$. Furthermore, the completely blank pattern p_0 is a numeral in base $k+rk$ which denotes the number 0, and pattern p_1 is a numeral in base $k+rk$ which denotes number k . Define: $d_e: Z^+ \rightarrow Z^+$, $d_e(v) = k$ if the expression in base h of v has exactly one coefficient equal to Y , $d_e(v) = 0$ if the expression in base h of v has exactly one coefficient equal to N , $d_e(v) = 0$ if the expression in base h of v has exactly one coefficient equal to N_{jii} for some j, i and u , $d_e(v) = v$ otherwise. Then, d_e is recursive and, if $d(v_e) = m_e$, $d_e(v) = m$. Therefore, d is a decoding of $P_F(B_\#)$ to $P_F(B_1)$ relative to e . Theorem 6 is thus proved q.e.d.

Chapter 3

DYNAMICAL PHENOMENA, EXPLANATIONS, AND THEORIES

1. Introduction
 2. Dynamical phenomena
 - 2.1 Magnitudes and dynamical models
 - 2.2 The correspondence between models and systems, and the concept of a dynamical phenomenon
 - 2.3 Dynamical studies as attempts to solve dynamical problems
 3. Frameworks and explanations
 - 3.1 Frameworks
 - 3.2 Dynamical explanations
 - 3.3 Explanations as solutions of dynamical problems, and the inductive method
 4. Principles, laws, and theories
 - 4.1 Theoretical principles, specific principles, and laws
 - 4.2 Theoretical frameworks and theoretical explanations
 - 4.3 Dynamical theories
 - 4.4 The deductive method for solving dynamical problems, and the heuristic value of dynamical theories
 5. Two examples of dynamical theories
 - 5.1 The theoretical principles of Impetus theory (or Aristotelian & Impetus dynamics)
 - 5.2 The standard models of Impetus theory
 - 5.3 The theoretical principles of Newtonian dynamics
 - 5.4 The standard models of Newtonian dynamics
 - 5.5 Two inconsistent theories which explain the same phenomena
 - 5.6 The heuristic values of Impetus theory and Newtonian dynamics
-

1. Introduction

In this chapter, I propose a theory of the explanation of a deterministic process.

This theory is based on a general view of scientific explanation according to which:

(i) explanations are solutions of *explanatory problems*; (ii) these problems

essentially consist in specifying a *model* of a system or process which we want to explain; (iii) explanatory problems come in different *types*; (iv) the *type of explanatory problem* depends on the *type of model* which we attempt to specify; (v) the *type of model* depends on the *type of mathematical structure* attributed to the system which we want to explain. In the first three sections, I articulate this general view for one particular type of problem, which I call a "*dynamical problem*". This problem consists in specifying a deterministic *dynamical model* of some system or process, and I propose to think of a *dynamical explanation* as a solution of a dynamical problem.

The main goal of the last two sections is to show that this theory of the explanation of a deterministic process also allows us to better understand the structure and function of *dynamical theories*. According to this view, a dynamical theory determines a set of *dynamical explanations*, and it also gives us *heuristic rules* for generating these explanations. I thus propose to think of the *heuristic value* of a dynamical theory as a rough measure of the efficiency of these rules. I finally suggest that the heuristic value of dynamical theories may play an important role in the choice between two competing theories.

This chapter analyzes a problem which is the main focus of many empirical sciences. This problem can be described as the attempt to *explain a dynamical phenomenon*. Some classic examples of dynamical phenomena are those which Galileo explained: free fall, the motion of a sphere on an inclined plane, projectile motion, etc. But dynamical phenomena are not limited to the domain of mechanics

or physics. Many other sciences try to explain phenomena of this kind. For example, demography is interested in the laws which govern the growth of a population under specified conditions. Chemistry may be interested in studying how the concentration of some substance varies during a certain reaction. Cognitive science attempts to explain the mental operations of a subject involved in some intellectual activity. Very probably, all sciences are interested in the explanation of some dynamical phenomenon. Obviously, this is not the only type of problem which an empirical science deals with. Nevertheless, if we were able to describe the logical structure of this type of explanation, we could demonstrate a deep structural identity between scientific disciplines which, at a more superficial level, may seem worlds apart.

I take "scientific explanation" to properly refer to those explanations which scientists attempt to produce as the result of their empirical or theoretical research. What I intend is thus best expressed by the words "explanations of *phenomena*". To have an idea of what qualifies as a scientific explanation in this strict sense, and what does not, it is best to start with some typical examples. The simplest examples I can think of are Galileo's explanations of *free fall*, *projectile motion*, and *the motion of a sphere on an inclined plane*. Another typical example is the explanation of *planetary motion* provided by a model of classical mechanics. On the other hand, most of the concrete examples discussed in the literature on scientific explanation *do not* qualify as scientific explanations in this strict sense, for they are not explanations of *phenomena*. This literature has been mostly

concerned with the explanation of *singular events*. But, in the first place, singular events are not phenomena, for phenomena are organized complexes of events. In the second place, the explanation of singular events (if it is scientific in a derivative sense) *presupposes* the explanation of the phenomena of which those events are parts. The second favorite topic of the literature on scientific explanation is the explanation of a *law*. These are not scientific explanations in the strict sense either, for they are not explanations of phenomena. The reason, again, is that a phenomenon is not a law but, rather, a system or process which can be described by means of an organized complex of laws.

The problem of scientific explanation is usually studied from a very general point of view. The standard approach starts by specifying a set of conditions which an empirical theory should satisfy, and then asks how a theory of this kind could produce explanations of specific phenomena¹. As a consequence, this approach has not paid much attention to the specific structure of the phenomena which scientists attempt to explain. I believe that the general problem of scientific

¹ See for example Hempel (1965), Nagel (1961), and Braithwaite (1964). A good introduction to the huge literature on scientific explanation is Salmon (1990). For some of the most recent contributions see Kitcher and Salmon (1989). Even those approaches which reject the received view of explanation still maintain that *scientific* explanation can only be understood *after* we analyze the structure of empirical theories. See, for example, van Fraassen (1980) and Suppe (1989). Also notice that the doctrine of the priority of the analysis of scientific theories does not depend on the choice of a particular solution to this problem. In fact, the received view of explanation presupposes a *syntactic* analysis of theories, while van Fraassen (1970, 1972, 1980, 1989) and Suppe (1974, 1989) subscribe to different versions of the *semantic* view. Other versions of the semantic view can be found in Suppes (1957, 1967, 1969), Przelecki (1969), Sneed (1971), Wessels (1974), Stegmüller (1976, 1979), Dalla Chiara and Toraldo di Francia (1981), Baltzer, Moulines, and Sneed (1987), Giere (1984, 1985, 1988). None of these analyses is completely satisfactory, but their detailed criticism goes beyond the scope of the present work. See, however, sections 2.2, 2.3, 4.3, and 4.4.

explanation can only be attacked by recognizing that *all scientific explanations* (in the strict sense of this term) *are explanations of specific types of phenomena*. In other words, to study scientific explanation in general -- without reference to the specific form of the phenomena which are to be explained -- is not a good strategy. In fact, *each specific type of phenomenon has a specific mathematical structure, and individuating this structure is a necessary condition for understanding the specific form of the explanations of that type of phenomenon*.

This may sound as an extremely strong form of Platonism but, in fact, it is not a metaphysical claim. It is rather a *methodological* claim about the structure of a typical *problem* which science attempts to solve. By definition, a system, object, entity, or process π is a phenomenon of type Π_{Δ} if it belongs to a certain subset Π_{Δ} of an appropriate set Δ of mathematical structures. My methodological point, then, is that science attempts to construct *explanations of specific types of phenomena*. What this means is that these explanations are *solutions of a special kind of problem*. A constitutive part of this problem is the *hypothesis* that a concrete process ψ (the explanandum) belongs to a specific phenomenon-type Π_{Δ} , that is, the hypothesis that ψ is a *phenomenon of type Π_{Δ}* . In other words, my methodological claim is that science attempts to explain concrete systems which *are assumed to be* phenomena of a specific type, and thus are *assumed to have a specific mathematical structure*. Science does not attempt to explain unstructured, or bare, objects. Therefore, if we are to understand scientific explanation, we'd better understand first the specific mathematical structure of the

phenomena which science attempts to explain.

Therefore, this chapter is not intended to discuss all scientific explanations, let alone the general nature of explanation. Rather, its scope is limited to a specific type of scientific explanations, that is, *explanations of dynamical phenomena*. Even though the literature on explanation is huge, this particular problem has been neglected so far². Dynamical phenomena obviously are an important type of scientific explanandum, but not all explananda are of this kind. For example, statistical phenomena certainly have a different structure. To elaborate an adequate classification of the different types of phenomena is not a trivial matter, and I believe that this is one of the primary goals of the philosophy of science. I also believe that this goal can only be achieved by employing a piece-meal strategy. That is, we need to individuate specific types of phenomena, and then to make explicit the mathematical structure proper of each type, and the structure of the explanations adequate for that type. This chapter is thus a first step towards the achievement of this general goal. In other words, I believe that dynamical phenomena are one of the types which will be included in a broader classification of phenomena.

Instead of starting from a general view of the structure of empirical theories,

² Van Fraassen's early semantic view of *theories* employs a dynamical approach (van Fraassen 1970, 1972; Wessels 1974). However, his analysis of *scientific explanation* is extremely general. Very briefly, van Fraassen (1980) provides two general theories. The first tells us what an *empirical theory* is, the second what an *explanation* is, and scientific explanations are explanations which use scientific theories in a certain way. More precisely, all scientific explanations are answers to why questions which draw upon scientific theories to provide the requested information. No attempt is made to understand the specific structure of the explanations of *dynamical phenomena*.

I start from the analysis of *dynamical phenomena*. This analysis will then shed light on the concrete scientific *explanations* of such phenomena. Eventually, we will also be able to understand the structure of those *empirical theories* which subsume these explanations under general principles, namely *dynamical theories*. The conceptual framework which allows this inversion of perspective is *dynamical system theory*. Very briefly, we can study the explanations of dynamical phenomena independently from empirical theories because dynamical system theory permits us to understand *the basic mathematical structure* which is constitutive of any dynamical phenomenon. In fact, this mathematical structure turns out to be a dynamical system³.

My *general* view of scientific explanation is that *all scientific explanations*⁴ are *explanations of specific types of phenomena*, and that *explanations of phenomena are solutions of a special kind of problem*. In general, this problem has three components, and it can thus be identified with an ordered triple $\langle \psi, H_{\Delta}, F \rangle$. The first element ψ is the concrete system or process which we want to explain (that is, the *explanandum*); the second element H_{Δ} is the *hypothesis* of the problem, which states that ψ belongs to a certain *phenomenon-type* Π_{Δ} ; the third element F is the *request* of the problem, whose general form is: *find a specification of a model of ψ* . The thrust of this chapter is to show how this general view of scientific

³ Recall that a dynamical system is a structure $\langle T, M, \{g^i\} \rangle$ which satisfies: T is a set which represents time, and $T =$ reals, rationals, or integers (or the non-negative parts of these structures); M is the set of all complete states of the system (the *phase space*); $g^i: M \rightarrow M$, $g^0(x) = x$, and $g^{t+w}(x) = g^t(g^w(x))$. See section 2 of chapter 1 and, in particular, definition 1.

⁴ In the strict sense of this term.

explanation can be developed in detail for *one* particular type of phenomena, that is, *dynamical phenomena*.

The concept of problem which I use throughout this chapter is a special case of what Newell and Simon call the *set representation of a problem* (1972, 73-75). Newell and Simon's concept is in turn a special case of a more general set theoretical definition which has been recently proposed by Veloso (1984). Veloso's definition expresses in a rigorous way Polya's conception of problems (1945, 1962, 1965), and it is the foundation of an elegant mathematical treatment of the concepts of problem-reduction and problem-decomposition. Nickles (1978, 1980, 1981) has proposed a concept of problem which is essentially equivalent to Newell and Simon's.

Let me now briefly sketch the content of each section. The main goal of the next section is to make explicit the *mathematical structure* which is shared by all dynamical phenomena. Intuitively, a *dynamical phenomenon is a deterministic process or system ψ which can be truly described by means of a finite number of interdependent magnitudes*. These magnitudes, together with their time evolution functions, can thus be thought as a *dynamical model* of the system ψ ⁵. Some of these magnitudes are observable, others may not be so, but the values of all magnitudes are determined if the evolution functions and the initial conditions are known. This analysis does not presuppose special properties of the evolution

⁵ For example, consider the free fall of a body. Then, the *Galilean model of free fall* can be identified with the magnitudes position and velocity together with their time evolution functions defined by $Y[y, v](t) = y + vt + 1/2ct^2$ and $V[y, v](t) = v + ct$ (where y and v are, respectively, the initial position and velocity of the falling body, and c is a constant).

functions, such as continuity or differentiability, and I also treat the magnitude *time* very generally. As a special case, time is allowed to be *discrete*. An interesting consequence of this approach is that *computational* models turn out to be a special type of dynamical models⁶.

In section 2.3, I introduce the concept of a *dynamical problem*. This is the crucial step of my analysis, for *I will later identify a dynamical explanation with a solution of a dynamical problem*. A dynamical problem has three components. First, a *concrete system or process* ψ which we want to explain; second, the *hypothesis* H , which states that ψ is a dynamical phenomenon; third, the *request* F , which asks of finding a specification of a dynamical model of ψ . Therefore, a solution of an arbitrary dynamical problem $\langle \psi H F \rangle$ is a *specification of a dynamical model of ψ* . I have mentioned above that a dynamical model can be thought as a finite number of *magnitudes* together with their *time evolution functions*. A dynamical model thus is an abstract entity. However, this abstract entity can always be linguistically specified by means of a finite number of mathematical formulas which express the time evolution functions of each magnitude⁷. A solution of a dynamical problem $\langle \psi H F \rangle$ will thus in general have two components: a *linguistic part*, and a *dynamical model of ψ* which is specified or determined by the linguistic part.

⁶ In particular, computational models of *cognition* can thus be thought as models of a special type of dynamical phenomena.

⁷ I require that the time evolution functions of an arbitrary dynamical model be *definable* by means of some mathematical operators.

These considerations thus introduce the main topic of section 3, where I finally make explicit my view of the explanation of dynamical phenomena. Intuitively, *explaining a dynamical phenomenon consists in specifying one of its dynamical models*. This means that we must find a finite number of mathematical formulas which express the time evolution functions of the magnitudes constitutive of one of these models. Usually, however, these mathematical formulas are not determined directly. Instead, one explicitly states some *hypotheses* from which the evolution laws of each magnitude can be deduced. For example, Newton's principle of universal gravitation and the second principle of dynamics (together with other specific assumptions) allow us to deduce the position and velocity laws of a planet which revolves about the sun. I thus propose to think of a *dynamical explanation of a system ψ* as a structure with two components. The first element is a *set of hypotheses*, and the second element is a *dynamical model of ψ whose evolution laws can be deduced from the hypotheses*. The set of hypotheses thus specifies or determines the dynamical model, so that *any dynamical explanation can also be thought as a solution of a dynamical problem*. To illustrate these ideas, I consider four classic examples of scientific explanations. In particular, I show that Galileo's explanation of free fall conforms to this model. A second example is the explanation of free fall which can be formulated in the context of Impetus theory. I show that my analysis also applies to this case which, at first glance, might seem of a different kind.

The definition of a dynamical explanation which I propose in section 3 is quite

general, in the sense that it does not impose any special condition on the set of hypotheses which specify a dynamical model of a given system ψ . However, the hypotheses which are usually employed in *theoretical explanations* have a special form. The goal of section 4.1 is to analyze the specific form of these hypotheses. In particular, I focus on *theoretical principles*. Intuitively, a theoretical principle is an equation which expresses the time evolution function of a magnitude as a mathematical function of the time evolution functions of other magnitudes. A typical example is the second principle of dynamics $\mathbf{A}(t) = \mathbf{F}(t) / \mathbf{M}(t)$, where $\mathbf{A}(t)$, $\mathbf{F}(t)$, and $\mathbf{M}(t)$ are the time evolution functions of the acceleration, force, and mass of an *arbitrary* body. We thus see that a theoretical principle holds for some specified *types* of time evolution functions. In this sense, a theoretical principle is *general*. Other assumptions employed in theoretical explanations, instead, are *specific*, for they are supposed to hold only for the particular system or process which we want to explain.

These considerations thus introduce the main topic of section 4, where I propose a new view of the structure of *dynamical theories*. In its simplest form, a *dynamical theory* is composed of three elements. The first element, the domain of the theory, is a *set of processes or systems* which the theory intends to explain. The second element is a *set of theoretical principles*, and the third element is a *set of theoretical explanations* of some of the systems in the domain of the theory. In section 4.3, I express this general view of dynamical theories in a precise form, and I then compare it to the syntactic and to the semantic approach. My basic

point is that either approach highlights one essential component of dynamical theories, namely, the *linguistic component* and the *model theoretic component*. However, dynamical theories cannot be *reduced* to either component.

The main thesis of section 4.4 is that some dynamical theories are extremely useful tools for *generating* or *producing* dynamical explanations. In other words, I maintain that some *dynamical theories allow us to efficiently solve dynamical problems*. This property of dynamical theories depends on three different sets of *heuristic rules* which are respectively associated with three different components of a dynamical theory. I thus propose to think of the *heuristic value* of a dynamical theory as a rough measure of the power of these three sets of heuristic rules.

To illustrate these ideas, I then consider two examples of dynamical theories. The first theory, which I call "Impetus theory", expresses some of the basic intuitions of Aristotelian dynamics, conjoined with some ideas taken from Impetus dynamics. The second theory, which I call "Newtonian dynamics", is the fragment of classical mechanics whose only theoretical principles are the second principle of dynamics, and the definitions of velocity and acceleration. The main thrust of section 5 is to show that my analysis of dynamical theories allows us to understand the *conceptual relations* between these two theories, and that it is possible to give a detailed *story* which explains how Newtonian dynamics *could have been produced* starting from Impetus theory. These considerations are based on a quite surprising theorem which I prove in section 5.5. This theorem affirms that Newtonian dynamics and Impetus theory are *translatable* even though,

under the natural interpretation which identifies the concepts of total force of the two theories, they are *inconsistent*. What makes the translation possible is in fact an *unnatural* interpretation which identifies the Newtonian total force with the propulsive force of Impetus theory.

Finally, in section 5.6, I compare the *heuristic values* of Newtonian dynamics and Impetus theory, and I conclude that, even though the two theories are translatable, Newtonian dynamics is a much more useful tool for representing and solving dynamical problems.

2. Dynamical phenomena

Mutation or change is perhaps the most basic feature of our experience⁸. At any given time, we perceive the world as a collection of different objects which are constitutive of certain *magnitudes*⁹. These magnitudes assume different *values* at different times, and their values belong to exactly one of the objects which are constitutive of a specific magnitude. For example, consider position and force. Position usually has one constitutive object, while force often involves a pair of objects. We may in fact measure the position of a body x, or the force on a body y exerted by a body x. If x and y are two different objects, the two expressions "the magnitude position of x" and "the magnitude position of y" refer to two different magnitudes. The expression "the magnitude force on y exerted by x" refers to a third magnitude. The object x is constitutive of the first magnitude, y is constitutive of the second, and both x and y are constitutive of the third. Any magnitude determines a set of possible values and, at any given time, exactly one of these values belongs to one of the constitutive objects of that magnitude. An appropriate name for this special object is "the substance of a magnitude"¹⁰. The

⁸ The subject of this paragraph ("we") has a specific referent: that group of scientists who attempt to construct deterministic dynamical models of systems which evolve through time. What this paragraph attempts to convey is a general picture of motion which I believe is often implicit in this scientific discipline.

⁹ The sense in which a sequence of objects is constitutive of a magnitude is explained below.

¹⁰ This name is not intended to evoke *all* the meanings traditionally associated with the term "substance". However, this name is intended to evoke the idea of an entity which is constitutive of certain properties, and which is the subject to which these properties are attributed. Each value of a magnitude can be thought to determine a different property, and this property belongs to the substance of that magnitude. Furthermore, the substance of that magnitude is constitutive of each of these properties. For example, if y is the position of x, then x has the property of having y which

substance of a magnitude is thus that object which is constitutive of that particular magnitude, and to which the values of that magnitude belong. For example, the substance of the magnitude position of x is the body x , while the substance of the magnitude force on y exerted by z is the body y ¹¹. The values of position and force belong, respectively, to the bodies x and y . Consider now all the possible magnitudes. At any given time, exactly one value of each magnitude belongs to its substance. Therefore, for each magnitude, there is a function whose domain is time and whose codomain is the set of possible values of that magnitude. This function is the *time evolution*, or the *motion*, of that particular magnitude. Mutation or change can thus be identified with the set of all these functions.

An important feature of change is that the time evolution functions of some sets of magnitudes may form a closed system, so that they are independent from the motions of all other magnitudes. This means that, for any such set, the variation of any of its magnitudes may depend on the magnitudes which belong to the same set, but this variation does not depend on any other magnitude which is not in this set. *As a first approximation, we may identify the time evolution functions of all the magnitudes in each of these closed sets with a dynamical model.* In what follows, I will exclusively consider dynamical models which involve a *finite* number of

is the position of x . The substance of the magnitude position of x is x , x is constitutive of the property of having y which is the position of x , and this property belongs to x .

¹¹ Notice that also the body z is constitutive of the magnitude force on y exerted by z . Nevertheless, the substance of this magnitude is y , because the value of the magnitude force on y exerted by z belongs to y , not to z . It might seem that this value could belong to z as well. However, the force on y exerted by z is a component of the total force on y , it is not a component of the total force on z . This is why the choice of y is not arbitrary.

interdependent magnitudes¹². A typical example of a dynamical model is the Galilean model of free fall. This model only involves two interdependent magnitudes: the position and the velocity of a falling body. The time evolution of the velocity depends on the initial velocity alone, but the time evolution of the position depends on both the initial position and the initial velocity. Therefore, taken together, the position and the velocity of a falling body form a closed system¹³.

Any dynamical model has four basic properties. First, *some of its constitutive magnitudes can be observed, measured, or detected*. For example, if we consider again the Galilean model of free fall, all magnitudes are observable. In general, however, the observable magnitudes are a subset of all the magnitudes constitutive a dynamical model, and this subset always includes the magnitude time.

Second, *the time evolution functions of a dynamical model are related in such a way to generate a dynamical system*. In the first place, this means that the time evolution function of each magnitude may depend only upon the initial values of the (other) magnitudes constitutive of the model. Therefore, strictly speaking, each magnitude M_i is associated with a set of time evolution functions $\{f_i[x_1 \dots x_i \dots x_k](t)\}$,

¹² A generalization to infinite sets is possible, but it involves technical complications which are not necessary for my present purposes. The basic problem is that we want to be able to *specify* or *determine* the time evolution functions of each magnitude of an arbitrary dynamical model. If there were an infinite number of such functions, we should then require that: (1) all these functions be definable in some language; (2) that the set of all formulas which define these functions be (partially) recursive.

¹³ "Closed system" is not intended here in the usual sense of classical mechanics -- an isolated system whose internal forces only depend on the positions of its constitutive particles.

where x_i is the initial value of magnitude M_i . This implies that:

$$[1] \quad f_i[x_j \dots x_k](0) = x_i;$$

for example, in the case of the Galilean model of free fall, the time evolution functions of the position and velocity of a falling body are determined by the equations $Y[y, v](t) = y + vt + 1/2 ct^2$ and $V[y, v](t) = v + ct$, where c is a constant. These equations obviously imply $Y[y, v](0) = y$ and $V[y, v](0) = v$. In the second place, the time evolution function of each magnitude also satisfy the composition principle:

$$[2] \quad f_i[x_j \dots x_k](t+w) = f_i[f_j[x_j \dots x_k](t) \dots f_k[x_j \dots x_k](t)](w);$$

it is easy to verify that, for the Galilean model of free fall, $Y[y, v](t+w) = Y[Y[y, v](t), V[y, v](t)](w)$ and $V[y, v](t+w) = V[Y[y, v](t), V[y, v](t)](w)$. The important point is that, whenever the time evolution functions of a set of magnitudes satisfy conditions [1] and [2], it is possible to use these functions to define a dynamical system (see sec. 2.1, th. 1). In this sense, the magnitudes constitutive of a dynamical model form, induce, or *generate*, a dynamical system.

Third, *all the time evolution functions of a dynamical model can be specified, or determined, by means of some mathematical formula.* That is, if f_i is an arbitrary time evolution function, there is some mathematical formula α_i such that " $f_i[x_j \dots x_k](t) = \alpha_i$ " defines f_i .

Fourth, *for any magnitude which is not observable, the time evolution function of at least one observable depends on it.* In other words, non-observable magnitudes are allowed to be components of a dynamical model only if they

contribute to the observable behavior of the model. If they do not, we'd better eliminate them.

The goal of the next section is to formulate the previous observations about a *dynamical model* in a rigorous way. Intuitively, a *dynamical phenomenon* is a deterministic process or system which can be truly described by means of a finite number of interdependent magnitudes. I will then use the definition of a dynamical model which I give in the next section to express this idea in a precise manner. In fact, I will define a dynamical phenomenon as any system or process which is isomorphic to the dynamical system generated by some dynamical model (see section 2.2).

2.1 Magnitudes and dynamical models

A magnitude is usually referred to by means of a description of the form "the magnitude n of s relative to $o_1 \dots o_m$ ". We have already seen two examples of such descriptions: "the magnitude *position* of x " and "the magnitude *force* on y exerted by z ". Usually, " n " is the ordinary name of the magnitude, s is its substance, and $o_1 \dots o_m$ are the other objects constitutive of that magnitude. An arbitrary value of a magnitude is usually taken to be a number or, more generally, some object which belongs to the domain of a specified mathematical structure. For example, the values of the magnitude *force* on y exerted by z are three dimensional vectors. For our purposes it is convenient to think of a magnitude as having two components. First, the set of all possible values and, second, the

sequence $\langle "n" s o_1 \dots o_m \rangle$ which is used in the description of that magnitude. For example, the magnitude *force* on *y* exerted by *z* can simply be identified with $\langle R^3 \langle "force" y z \rangle \rangle$, where R^3 is a Euclidean three-dimensional vector space.

From the set theoretical point of view, each magnitude is thus an ordered pair $\langle M_i i \rangle$. The first element M_i is the set of all the possible values of the magnitude. This set must be the domain of some specified mathematical structure, and M_i is called "*the quantity*" of the magnitude. The second element *i* is a finite sequence $\langle a_1 \dots a_k \rangle$ which I call "*the label*" of the magnitude. This is expressed by the following definition:

Definition 1 (magnitude)

M_i is a magnitude iff:

$M_i = \langle M_i i \rangle$, where M_i is the domain of a specified mathematical structure, and *i* is a finite sequence¹⁴.

M_i is called the quantity of M_i , *i* is called the label of M_i , and each element $x \in M_i$ is called a value of the magnitude M_i .

We see that, by definition 1, a magnitude simply is a set of mathematical objects with a fixed label. This allows the same set to stand for different magnitudes, as it is customary in science. For example, the set of the real numbers, R , may represent time, position, velocity, etc. In the simplest case, we can think of the label of a magnitude as the sequence whose only element is the usual name of that magnitude. For example:

¹⁴ Notice that the symbol " M_i " in *italics* refers to the magnitude $\langle M_i i \rangle$, while the symbol " M_i " in normal font refers to the quantity of this magnitude.

time = $\langle R \langle \text{"time"} \rangle \rangle$;
horizontal position = $\langle R \langle \text{"horizontal position"} \rangle \rangle$;
vertical velocity = $\langle R \langle \text{"vertical velocity"} \rangle \rangle$, where R are the real numbers.

I have suggested above that a *dynamical model* can be identified with a finite number of interdependent magnitudes whose time evolution functions form, or generate, a dynamical system. Some of these magnitudes are observable and, for any magnitude which is not observable, there is at least one observable whose time evolution function depends on it. Furthermore, each time evolution function can be defined by means of some mathematical formula. All these requirements are expressed by the following definition:

Definition 2 (dynamical model)

P is a dynamical model iff:

$P = \langle \langle M_1 \dots M_n T \rangle \langle f_1 \dots f_n \rangle O \rangle$ and

- (1) each magnitude in $\langle M_1 \dots M_n T \rangle$ has a different label;
- (2) the magnitude T is time and its quantity T is one of the following: R, R^+, Q, Q^+, Z, Z^+ , where R are the real numbers, Q the rationals, Z the integers, and R^+ are the non-negative reals, Q^+ the non-negative rationals, and Z^+ the non-negative integers;
- (3) for any function f_i in $\langle f_1 \dots f_n \rangle$, $f_i: M_1 x_1 \dots x_n M_n x_n T \rightarrow M_i$ satisfies:
 - (a) $f_i[x_1 \dots x_n](0) = x_i$;
 - (b) $f_i[x_1 \dots x_n](t+w) = f_i[f_1[x_1 \dots x_n](t) \dots f_n[x_1 \dots x_n](t)](w)$;
 - (c) there is a function form¹⁵ α_i , such that $f_i[x_1 \dots x_n](t) = \alpha_i$;

¹⁵ Intuitively, a function form is a mathematical formula which can be used to define a the time evolution function of a magnitude. For example, the formula " $y + vt + 1/2ct^2$ " is a function form, for it can be used to define the function $Y[y v](t)$ (by writing a function term in italics, I refer to the function associated with that term, not to a value of that function). The reason why I require all the evolution functions of a dynamical model to be definable is that we want to be able to express, specify, or determine, these functions. For a precise characterization of the concept of a function form, see the text below example 2.1.

- (4) O is a set¹⁶ whose elements are T and at least one more magnitude in $\langle M_j \dots M_k T \rangle$ and, if magnitude M_i is in $\langle M_j \dots M_k \rangle$ and $M_i \notin O$, there is at least one magnitude $M_r \in O$, such that f_r depends¹⁷ on M_i (that is, for each component which is not observable there is at least one observable which depends on it).

If we look at this definition from the point of view of the semantic conception of theories, then a dynamical model is a special type of what is usually called a "theoretical model" (Giere 1984, 1985, 1988; van Fraassen 1980, 1989). The empirical substructure of a dynamical model consists of the time evolution functions of the observable magnitudes of that model. The set of all dynamical models is thus a theory in the semantic sense. We may call this theory "the theory of dynamics", or "dynamics".

Example 2.1 (The Galilean model of free fall)

Let T = time, Y = vertical position, and V = vertical velocity of a freely falling body. Let $Y[y v](t) =: y + vt + 1/2 ct^2$, and $V[y v](t) =: v + ct$, where y is the initial position of the falling body, v the initial velocity, and c is a constant. It is trivial to verify that $P = \langle\langle Y V T \rangle \langle Y[y v](t) V[y v](t) \rangle \{Y V T\}\rangle$ is a dynamical model¹⁸. I will call this dynamical model "*the Galilean model of free fall*".

¹⁶ Intuitively, O is the set of all the observable magnitudes of the dynamical model P . I take an observable magnitude to be *any* magnitude for which we have some measurement procedure. This, however, is not the only possible view about observables. What follows is compatible with any view which satisfies the minimal requirement that any observable magnitude be measurable or detectable.

¹⁷ The time evolution function f_r depends on magnitude M_i iff: there is $x_1 \dots x_i y_i \dots x_k t$ such that $f_r[x_1 \dots x_i \dots x_k](t) \neq f_r[x_1 \dots y_i \dots x_k](t)$.

¹⁸ By writing a function term in *italics* I refer to the *function* associated with that term, not to a value of that function. For example, $Y[y v](t)$ is the function $Y: YxVxT \rightarrow Y$ defined by $Y[y v](t) =: y + vt + 1/2ct^2$. Also, for notational convenience, I will often use *the same symbol* to refer to either the *quantity* of a magnitude, or to its time evolution *function*. For example, the first occurrence of "Y" in " $Y: YxVxT \rightarrow Y$ " refers to the time evolution function of the vertical position of a falling body, while the second and third occurrences refer to the quantity of this magnitude.

I have used the concept of a *function form* to formulate condition (3c) of definition 2. Intuitively, a function form is a mathematical formula which can be used to define the time evolution function of a magnitude. The concept of a function form can be made precise in the following way. First, let us stipulate that the variable "t" varies on the set T of all possible values of the magnitude time. Second, for each magnitude $\langle M_i \rangle$ different from time, let us introduce the individual variable x_i whose range is M_i (that is, the quantity of the magnitude), the function constant " f_i ", and all function terms of the form " $f_i[x_j \dots x_k](t)$ ", where all the variables $x_j \dots x_k$ may be missing. Third, let us assume that the function constant " f_i " expresses a function whose codomain is M_i , and whose domain is fixed by the context¹⁹. For instance, if " f_i " occurs in the function term " $f_i(t)$ ", the domain of f_i is T. However, if " f_i " occurs in the function term " $f_i[x_j \dots x_k](t)$ ", the domain of f_i is $M_j \times \dots \times M_k \times T$. Finally, let us specify the *mathematical operators* which are allowed to occur in an arbitrary *function form* α , and the *formation rules* which permit us to construct more complex forms from simpler ones, in such a way that no function constant " f_i ", or function term " $f_i[x_j \dots x_k](t)$ " occurs in any function form

*Furthermore, let us also require that any equation of the form " $f_i[x_j \dots x_k](t) = \alpha$ " be a syntactically correct definition of " $f_i[x_j \dots x_k](t)$ " (where any variable different from " x_j " ... " x_k " "t" which occurs in α is taken to be a constant).

The concept of a function form also allows us to define the concept of a law.

¹⁹ I refer to the function expressed by " f_i " in context " $f_i[x_j \dots x_k](t)$ ", by writing the context in *italics*. For example, $f_i(t)$ is the function expressed by " f_i " when " f_i " occurs in " $f_i(t)$ ".

A law is any equation of the form " $f_i(t) = \alpha$ ", where α is an arbitrary function form. For example, the *equations* " $Y(t) = y + vt + 1/2ct^2$ " and " $Y(t) = y + mvt/m + 1/2ct^2$ " are two equivalent *laws* of the position of a falling body. I will reserve the definite description "*the law ...*" for the equation which is customarily used to express that law. For example, the law of the position of a falling body is the equation " $Y(t) = y + vt + 1/2ct^2$ ".

If $P = \langle \langle M_j \dots M_k \ T \rangle \langle f_j \dots f_k \rangle \ O \rangle$ is a dynamical model, P naturally induces or *generates* a dynamical system $\pi(P) = \langle T \ M_j x \dots x M_k \ \{g^t\} \rangle$. What this means is the following. Each magnitude M_i can be thought to be a component of the phase space of $\pi(P)$, and the time advances of this dynamical system are defined by means of the time evolution functions $\langle f_j \dots f_k \rangle$ that is, $g^t(x_j \dots x_k) =: \langle f_j[x_j \dots x_k](t) \dots f_k[x_j \dots x_k](t) \rangle$. The precise definition of the system generated by a dynamical model is the following:

Definition 3 (the dynamical system generated by a dynamical model)

$\pi(P)$ is generated by P iff:

$P = \langle \langle M_j \dots M_k \ T \rangle \langle f_j \dots f_k \rangle \ O \rangle$ is a dynamical model, $\pi(P) = \langle T \ M_j x \dots x M_k \ \{g^t\} \rangle$, and $g^t: M_j x \dots x M_k \rightarrow M_j x \dots x M_k$ is defined by $g^t(x_j \dots x_k) =: \langle f_j[x_j \dots x_k](t) \dots f_k[x_j \dots x_k](t) \rangle$.

Notice that, by definition 3, for any dynamical model P , there is exactly one system $\pi(P)$ generated by P . It is easy to prove that $\pi(P)$ is a deterministic

dynamical system. Before proving this theorem²⁰, however, let us look at a simple example of a dynamical system generated by a dynamical model.

Example 3.1 (the dynamical system generated by the Galilean model of free fall)

Let P be the dynamical model defined in example 2.1. Then, $\langle T YxV \{g^t\} \rangle$ is the system generated by P , where $g^t(y v) = \langle Y[y v](t) V[y v](t) \rangle$, $Y[y v](t) = y + vt + 1/2ct^2$, and $V[y v](t) = v + ct$.

Theorem 1 (the system generated by a dynamical model is a deterministic dynamical system)

If $\pi(P)$ is the system generated by a dynamical model P , then $\pi(P)$ is a deterministic dynamical system²¹

proof:

let $P = \langle \langle M_j \dots M_k T \rangle \langle f_j \dots f_k \rangle O \rangle$ be a dynamical model, and let $\pi(P) = \langle T M_j \dots M_k \{g^t\} \rangle$ be the system generated by P . Then, by definition 3:

[1] for all $t \in T$, $g^t(x_j \dots x_k) =: \langle f_j[x_j \dots x_k](t) \dots f_k[x_j \dots x_k](t) \rangle$

Recall that $\pi(P)$ is a dynamical system iff

(a) $g^0(x_j \dots x_k) = \langle x_j \dots x_k \rangle$

(b) $g^{t+w}(x_j \dots x_k) = g^t(g^w(x_j \dots x_k))$

proof of (a):

from [1], and (3a) of definition 2:

1. $g^0(x_j \dots x_k) = \langle f_j[x_j \dots x_k](0) \dots f_k[x_j \dots x_k](0) \rangle = \langle x_j \dots x_k \rangle$ //q.e.d.//

proof of (b):

from [1], and (3b) of definition 2:

$$g^{t+w}(x_j \dots x_k) = \langle f_j[x_j \dots x_k](t+w) \dots f_k[x_j \dots x_k](t+w) \rangle = \langle f_j[x_j \dots x_k](w+t) \dots f_k[x_j \dots x_k](w+t) \rangle = \langle f_j[f_j[x_j \dots x_k](w) \dots f_k[x_j \dots x_k](w)](t) \dots f_k[f_j[x_j \dots x_k](w) \dots f_k[x_j \dots x_k](w)](t) \rangle =$$

²⁰ The proof of this theorem depends on conditions (3a) and (3b) of definition 2. These conditions not only are sufficient for a dynamical model to generate a dynamical system but, in a certain sense, they are also necessary. In fact, suppose we are given a structure $P = \langle \langle M_j \dots M_k T \rangle \langle f_j \dots f_k \rangle O \rangle$ which satisfies all conditions of definition 2 except (3a) or (3b). Then, it is easy to prove that the system $\pi(P) = \langle T M_j \dots M_k \{g^t\} \rangle$, where $g^t(x_j \dots x_k) =: \langle f_j[x_j \dots x_k](t) \dots f_k[x_j \dots x_k](t) \rangle$, is not a dynamical system. That is, if either condition (3a) or (3b) is not satisfied, the system generated by P is not a dynamical system.

²¹ Recall that $\langle T M \{g^t\} \rangle$ is a deterministic dynamical system iff: T is the reals, rationals, or integers (non-negative reals, non-negative rationals, or non-negative integers), M is a non-empty set, $g^t: M \rightarrow M$, $g^0(x) = x$, and $g^{t+w}(x) = g^t(g^w(x))$.

$g^t(f[x_j \dots x_k](w) \dots f_k[x_j \dots x_k](w)) = g^t(g^w(x_j \dots x_k))$ //q.e.d.//
 Theorem 1 is thus proved q.e.d.

2.2 The correspondence between models and systems, and the concept of a dynamical phenomenon

We have seen in the previous section that any dynamical model P generates a dynamical system $\pi(P)$. If we now consider an arbitrary system π which is isomorphic²² to $\pi(P)$, we can take P to be a dynamical model of π . I thus define:

Definition 4 (dynamical model of a system or process)

P is a dynamical model of π iff: P is a dynamical model, and π is isomorphic to the dynamical system $\pi(P)$ generated by P .

I take this definition to express in a precise way the intuitive idea that a dynamical model P *corresponds, or is similar in specified respects,* to a system π . The hypothesis " P is a dynamical model of π " is thus a particular type of what Giere calls "a theoretical hypothesis"²³. That is, I take an arbitrary theoretical

²² This is an isomorphism *between dynamical systems* or, in other words, a *dynamical isomorphism*. Recall that S_1 is (dynamically) isomorphic to S_2 just in case $S_1 = \langle T M_1 \{g^t\} \rangle$, $S_2 = \langle T M_2 \{h^t\} \rangle$, S_1 and S_2 are dynamical systems, and there is a bijection $f: M_2 \rightarrow M_1$ such that $f(h^t(x)) = g^t(f(x))$.

²³ See Giere (1984, 1985, 1988). To be completely honest to Giere, he maintains that a theoretical hypothesis should be of the form " P is similar to π in specified *respects* and to specified *degrees*" (Giere 1985, 80). I believe that (as far as dynamics is concerned) " P is a dynamical model of π " is the correct explication for " P is similar to π in specified respects". I do not claim that this explication also captures the intended meaning of the clause "to specified degrees". In fact, I maintain that this clause should not be included in the formulation of a theoretical hypothesis. Rather, I maintain that " P is similar to π to specified degrees" should be construed as a separate hypothesis that states the empirical adequacy of P . That is, I take " P is similar to π to specified degrees" to be equivalent to " P is an *empirically adequate* model of π ". What this means is that P is a dynamical model and, for each magnitude M_i in P , its evolution law " $f_i(t) = \alpha_i$ " is consistent with all the (possible) measurements of that magnitude.

hypothesis of dynamics to have the form "P is a dynamical model of π ". This hypothesis affirms that *the dynamical system generated by P and the system π have the same dynamical properties*²⁴. In this sense, the model P is similar to the system π as far as *the purely dynamical properties* of $\pi(P)$ are concerned. These are all those properties which exclusively depend on the dynamical (semi)group $\{g^t\}$ which is associated with P. More precisely, the properties which depend on the (semi)group $\{g^t\}$ which is constitutive of the dynamical system $\pi(P) = \langle T M \{g^t\} \rangle$ generated by P. Any other property of this system need not be shared by π . For example, if $\pi(P)$ has a topological structure, its topological properties need not be preserved, for I am *not* requiring the isomorphism between $\pi(P)$ and π to be a homeomorphism. In fact, π might not be a topological space at all.

Intuitively, *a dynamical phenomenon is a deterministic process which can be truly described by means of a finite number of interdependent magnitudes*. We can finally make this idea precise by identifying a dynamical phenomenon with any system or process which has some dynamical model. I thus define:

²⁴ A dynamical property is, by definition, any property of a dynamical system which is shared by all the dynamical systems isomorphic to that system. That is, **P** is a dynamical property of S iff: S is a dynamical system, S has property **P** and, for any system S[#] such that S is isomorphic to S[#], S[#] has property **P**. Therefore, if a system has a dynamical property, it is a dynamical system, and any two dynamical systems which are isomorphic have exactly the same dynamical properties. Conversely, if two systems have exactly the same dynamical properties, they must be isomorphic. Suppose they are not. Then, system S₁ has the property **P**₁ = *being isomorphic to S₁*. Since system S₂ is not isomorphic to S₁, S₂ has not property **P**₁. However, **P**₁ is a dynamical property of S₁. Therefore, since by assumption S₁ and S₂ have the same dynamical properties, S₂ has property **P**₁.

Definition 5 (dynamical phenomenon)

π is a dynamical phenomenon iff:
there is P such that P is a dynamical model of π .

By definition 5, a dynamical phenomenon π can be identified (*up to an isomorphism*) with the dynamical system $\pi(P)$ generated by *any* P which is a dynamical model of π . It is very often *convenient* to take $\pi = \pi(P)$ for some P, so that we can *talk* of the components of the total states of π . However, it should be kept in mind that, in general, π and $\pi(P)$ are *different* systems, which are guaranteed to be identical *only* with respect to their *dynamical properties*.

The class of all dynamical phenomena is not empty, for definition 4 implies that the system $\pi(P)$ generated by an arbitrary dynamical model P is a dynamical phenomenon. Therefore, the class of all dynamical phenomena will in general contain *abstract* systems. This class may however contain *concrete* systems as well²⁵. We can thus identify an *actual* dynamical phenomenon with any *concrete system or process* which is a dynamical phenomenon. More precisely, ψ is an actual dynamical phenomenon just in case ψ is a *concrete* object, entity, system,

²⁵ I take a *physical system* to be any system or process which might be an object of physical study. Therefore, some, *but not all*, physical systems are concrete systems. I take *concrete systems* to satisfy the following conditions. First, I assume that there are basic concrete systems, and that all basic concrete systems are physical systems. Second, a system ψ is concrete iff ψ is a basic concrete system, or there is some basic concrete system which realizes ψ . An *abstract system* is any system which is not concrete. The analysis of the realization relation (for the case of dynamical systems) is in ch. 1, def. 5 and def. 6. This relation is a quasi-ordering (reflexive and transitive).

or process, and ψ is a dynamical phenomenon²⁶.

Example 4.1 (the phenomenon of free fall)

Suppose ψ is the free fall of a body²⁷. If this *concrete* process ψ is identical to some dynamical phenomenon, then ψ is an *actual* dynamical phenomenon. If we take the Galilean laws for position and velocity to be empirically true, then we can take ψ to be isomorphic to the dynamical system $\psi(P)$ generated by P , where P is the dynamical model determined by those laws (see examples 2.1 and 3.1). In other words, we can take P to be a dynamical model of ψ , and ψ itself to be an actual dynamical phenomenon²⁸.

2.3 Dynamical studies as attempts to solve dynamical problems

My basic methodological postulate is that the objects which are studied by the

²⁶ Equivalently, ψ is an actual dynamical phenomenon just in case ψ is a concrete system, and the hypothesis " ψ is a dynamical phenomenon" is true. Obviously, this hypothesis might very well be false, namely if ψ is not a deterministic process, or if ψ cannot be truly described by means of a finite number of interdependent magnitudes (therefore, if determinism is false, the class of all actual dynamical phenomena is empty). The important methodological point, however, is that for many concrete processes, there are concrete epistemic contexts in which we take this hypothesis for granted. I will discuss this point below.

²⁷ By "free fall of a body", I mean the motion of a medium sized body released from any initial position in a small specified region on earth, with any initial vertical velocity.

²⁸ An evolution law " $f_i(t) = \alpha_i$ " is *empirically true* just in case it is consistent with any (possible) measurement of the magnitude M_i . If we assume the precision of a possible measurement of time, position, and velocity to be within appropriate limits, then we may take the Galilean laws of free fall to be empirically true. This concept of empirical truth is similar to the concept of "physical truth" proposed by Dalla Chiara and Toraldo di Francia (1981). However, physical truth is more general than my empirical truth, for my concept only applies to *evolution laws*, that is, formulas of the form " $f_i(t) = \alpha_i$ ". Furthermore, I believe that the concept of physical truth (and also my empirical truth) does not capture all the relevant usages of the term "true" in the physical sciences.

A dynamical model P is an *empirically adequate* model of ψ just in case all its laws are empirically true. I do not claim that accepting the hypothesis " P is an empirically adequate model of ψ " is sufficient for accepting the stronger " P is a dynamical model of ψ ". I only claim that it is necessary. However, I claim that, given the empirical adequacy of a model P , there may be concrete epistemic contexts in which also the acceptance of " P is a dynamical model of ψ " is justified. I take this to be historically confirmed. Galileo maintained that his laws were *true* (not just empirically true) and, if we look at *his concrete epistemic context*, his claim was fully justified (obviously, I am thinking of "justification" in a sufficiently broad sense, in which *values* play an important role).

science of dynamics are better seen as concrete systems which are *taken* to be identical to some dynamical phenomenon. My basic tenet is that *the dynamical study of a concrete process ψ presupposes the (implicit) identification of this process with some dynamical phenomenon π* , that is, with a dynamical system which can be described by means of a finite number of interdependent magnitudes. In other words, I claim that *the science of dynamics studies concrete processes (implicitly) identified with a specific type of mathematical structure. The science of dynamics does not study bare, or unstructured, objects*²⁹.

If this is true, *we can think of any concrete dynamical study as the attempt to solve a specific type of problem*. This problem involves three components. First, a concrete system or process ψ which is *the object of our study*. Second, *the hypothesis* of the problem: " ψ is a dynamical phenomenon". Third, *the request* of the problem, whose form is "find a specification of a dynamical model of ψ ". We can thus represent this problem as an ordered triple $\langle \psi \ H \ F \rangle$, where ψ , H , and F are, respectively, the object, the hypothesis, and the request. The hypothesis of the problem entails that ψ has some dynamical model, so that, *if this hypothesis is true*, the request of the problem can be satisfied³⁰. I will call any problem of

²⁹ We have seen that this mathematical structure is a *dynamical system*. Dynamical systems have a quite simple structure, for they are essentially groups. If time is restricted to the non-negative reals, rationals, or integers, then the set $\{g^t\}$ of all t -advance of $\pi = \langle T \ M \ \{g^t\} \rangle$ is a commutative monoid with respect to the composition operation. In other words, $\{g^t\}$ is a commutative semigroup with unity, where the unity is $g^0 =$ the identity function on M . Otherwise, $\{g^t\}$ is a commutative group.

³⁰ Notice that also a *specification* of a model of ψ must exist, for any dynamical model can be specified by listing: (1) a finite sequence of magnitudes, (2) a definition for each time evolution function, and (3) the observable magnitudes of the model. Condition (3c) of the definition of a dynamical model ensures that each evolution function $f[x_1, \dots, x_n](t)$ is definable.

the form $\langle \psi \ H \ F \rangle$ a dynamical problem.

The goal of any concrete dynamical study is thus to solve a particular dynamical problem $\langle \psi \ H \ F \rangle$. This goal is accomplished if we are able to *specify a dynamical model* of the given process ψ . The interesting fact about this problem is that a solution exists just in case the hypothesis of the problem H is true. This implies that this problem might in fact be unsolvable³¹. However, to start with, we simply disregard this possibility or, in other words, **(i)** we just *assume* H to be true³². What we do next is **(ii)** *specify* some dynamical model P , and then **(iii)** produce *empirical* evidence and arguments which make us (provisionally) accept the *specific* hypothesis " P is a dynamical model of ψ "³³. Obviously, if the epistemic context changes, the new evidence and arguments may prompt us to withdraw our claim. We thus recognize that what we believed to be true is 'in fact' false³⁴. Furthermore, if we fail to accept *any* model which we come up with, then

³¹ If the hypothesis of the problem H is false, then the definition of a dynamical phenomenon implies that the set of dynamical models of ψ is empty, so that the request of the problem cannot be satisfied.

³² If, as far as we know, it is *reasonable* to assume that ψ is a deterministic process which can be described by means of a finite number of interdependent magnitudes (recall that $H = "$ ψ is a dynamical phenomenon"). Any stronger initial justification for assuming H is neither necessary nor desirable. See the next footnote to understand why.

³³ Notice that this specific hypothesis *entails* the hypothesis of the problem H . This is why, to start with, we may just take H for granted. If, later on, we come up with a model which we accept as a model of ψ , our 'bold' initial assumption will be vindicated. On the other hand, if we are not able to accept any model which we have come up with, *at that point* we may have sufficient grounds to reject H and, thus, abandon the problem which we 'know' is unsolvable.

³⁴ This involves two things: (1) we recognize that the specific hypothesis " P is a dynamical model of ψ " is false and, consequently, (2) we recognize that the model P is a false description of ψ (that is, that the system $\psi(P)$ generated by P is not isomorphic to ψ).

we may even (provisionally) conclude that, after all, the concrete process ψ is not a dynamical phenomenon. If this is the case, we recognize that the hypothesis of the problem H is 'in fact' false, so that the problem 'is' unsolvable and we (provisionally) abandon it³⁵.

Even if a concrete process ψ were a dynamical phenomenon, and we were able to specify one of its models P, we might not be able to *know* or *recognize* that P is a model of ψ . The problem of understanding the conditions under which we accept or reject a specific hypothesis of the form "P is a dynamical model of ψ " is very important, but I will not explicitly address it. The reason is that I am now interested in establishing the general methodological framework in which epistemological questions should be posed. The elaboration of a detailed epistemology compatible with this framework goes beyond the scope of this chapter. It is, however, clear that this epistemology must be some form of fallibilism.

³⁵ Let me stress that this paragraph is only intended to convey a *general picture* of the process through which we attempt to solve a specific dynamical problem $\langle \psi \text{ H F} \rangle$. In particular, much more should be said about the (provisional) acceptance or rejection of a specific hypothesis of the form "P is a dynamical model of ψ ". A necessary condition for accepting this hypothesis is that we accept the weaker hypothesis "P is an empirically adequate model of ψ ", where P is an empirically adequate model of ψ just in case, for any magnitude M_i in P, its evolution law " $f_i(t) = \alpha_i$ " is consistent with any (possible) measurement of that magnitude.

It is also interesting to notice that a dynamical study is itself a dynamical process, so that we should be able to specify a dynamical model of a dynamical study. This implies that *the methodology of dynamics is itself a branch of dynamics*. The observations of this paragraph can thus be interpreted as an attempt at specifying the general features that a detailed *dynamical model* of a dynamical study is likely to have. Also see sections 3.3 and 4.4, where I propose two more detailed models of a dynamical study: the *inductive* model, and the *deductive* model. Both models are special cases of the general model which I have sketched in this paragraph.

One might wonder whether the acceptance of a specific hypothesis "P is a dynamical model of ψ " could ever be justified. If we only consider empirical evidence, we may at most infer that all the evolution laws of the magnitudes in P are consistent with all the measurements, and this is obviously not enough to conclude "P is a dynamical model of ψ ". There are many different approaches for dealing with this problem. The empiricist approach recommends that we weaken the specific hypothesis so that this weaker form can be empirically justified. The correct form of the specific hypothesis would thus be: "P is an *empirically adequate* model of ψ ". What this means is that, for each magnitude M_i in P, its law " $f_i(t) = \alpha_i$ " is consistent with all the (possible) measurements of that magnitude. I believe that this form of the specific hypothesis can be empirically justified³⁶. However, I disagree with the empiricist's further proposal that we should never accept anything more than "P is an *empirically adequate* model of ψ ". My basic reason depends on the observation that, in concrete epistemic contexts, scientists do accept hypotheses of the form "P is a dynamical model of ψ ", and that, when they do this, they *mean what they say*. What they mean is that *the dynamical properties determined by the model P are, exactly, the dynamical properties of ψ* or, more precisely, that *ψ is isomorphic to the dynamical system $\psi(P)$ generated by P*. The real epistemological problem is thus to *explain* how, in specific epistemic contexts, *this claim* is accepted or rejected. Giere has recently argued that a decision theoretic approach may adequately explain how realistic

³⁶ If we only consider *actual* measurements this is obvious, but I believe that some form of inductive justification can be devised even if all *possible* measurements are considered.

hypotheses are accepted in concrete epistemic contexts (1985, 1988). In general, a realistic hypothesis affirms the existence of a *specific isomorphism* between a structure generated by a model and a concrete system. Therefore, "P is a dynamical model of ψ " is a special type of realistic hypothesis. Giere's decision theoretic approach might thus explain how a hypothesis of this type is concretely accepted or rejected.

Let me also stress that accepting the specific hypothesis "P is a dynamical model of ψ " does not entail any *strong* realistic commitment. In particular, nothing follows about the reality of theoretical magnitudes which may occur in P (like, for example, mass or force). On the other hand, no 'empiricist' commitment follows either, for we cannot assert the reality of *any* magnitude, be it theoretical or not. What follows is that the concrete process ψ is a *deterministic dynamical system* (essentially a group structure) which is *isomorphic* to the dynamical system $\psi(P)$ generated by P. However, even though the system ψ can be truly described *as if* it were composed by all the magnitudes which are components of the dynamical system $\psi(P)$, none of these magnitudes must be constitutive of ψ ³⁷. Finally, a (weak) commitment to the existence of modalities does follow, for "P is a dynamical model of ψ " entails " ψ is a dynamical system", and any such system

³⁷ The isomorphism between $\psi(P) = \langle T M_1 \{g^i\} \rangle$ and $\psi = \langle T M_2 \{h^i\} \rangle$ only preserves the *purely dynamical properties* of $\psi(P)$, that is, those properties which exclusively depend on the dynamical (semi)group $\{g^i\}$. Therefore, if the hypothesis "P is a dynamical model of ψ " is true, $\psi(P)$ and ψ are guaranteed to be similar *only* with respect to their purely dynamical properties. Any other property that $\psi(P)$ may have need not be preserved. In particular, the property of having components is not a dynamical property of $\psi(P)$, because it is always possible to find a system isomorphic to $\psi(P)$ which does not have components. Therefore, "P is a dynamical model of ψ " does not entail anything about the reality of *any* component magnitude of P (be it theoretical or not).

has a simple modal structure³⁸. Therefore, the philosophical position implied by my view of dynamics is a weak form of constructive realism (where weak modifies "realism").

From the epistemological point of view, this position is *realistic* in three different senses: (1) the existence of concrete *systems or processes* is assumed; (2) concrete systems are attributed a *modal structure*; (3) concrete systems are attributed *specific dynamical properties*. It is *constructive* because the *attribution* of a modal structure and of *specific* dynamical properties depends on the acceptance of a hypothesis of the form "P is a dynamical model of ψ ". In other words, modal structure and *specific* dynamical properties are not *initial* features of the concrete systems. Rather, they are the *results* of our successful scientific inquiries. From these remarks, it should be quite clear that the appropriate name for this philosophical position is "*dialectical realism*". Perhaps, this needs to be better explained. For any concrete system ψ , either ψ is a dynamical phenomenon, or it is not. If it is, then ψ objectively has a modal structure and some dynamical properties. If it is not, ψ objectively lacks a modal structure or dynamical properties. Therefore, from the *ontological* point of view, this philosophical position is realistic in the usual sense. However, from the *epistemological* point of view, it is *dialectical*, for what we *know* about the modal structure and the specific dynamical properties of a particular system ψ is the result of our attempts to solve a *concrete problem*. Furthermore, our knowledge

³⁸ See Giere (1985) for an interesting discussion of modalities in the context of the debate between empiricist and realist positions.

may change with time, and what we know at a later time is typically *inconsistent* with what we knew at previous times.

The definition of a dynamical model implies that a system may have models with some component which is not an observable magnitude. It is important to notice that, if we have a theory³⁹, the theory may tell us what these additional magnitudes are. For example, according to classical mechanics, the total states of any mechanical system can be identified with the values of *position* and *momentum*. This means that any mechanical system has at least one dynamical model whose only components are position and momentum⁴⁰. Some authors do not consider momentum to be observable. If we take this view, then classical mechanics tells us that any mechanical system has at least one model with a non-observable component, namely momentum. It should also be noticed, however, that many mechanical systems also have models whose components are all observable. In fact, if the mass is constant, a mechanical system has a model

³⁹ By "theory" I mean something sufficiently similar to what we take to be our best examples of theories. One of these typical examples is classical mechanics. If we agree on this intended meaning (which I take to be the usual one), then I believe that the semantic view is too general to provide an adequate explication of this term. This is clear when we realize that the set of all dynamical models is a theory in the semantic sense. Obviously, this set of models is not a theory in the usual sense.

⁴⁰ This can also be taken as a definition of a "mechanical system": π is a mechanical system iff π has some dynamical model whose only components are positions and momenta. This definition (together with the definition of dynamical phenomenon) implies that any mechanical system is a dynamical phenomenon. Furthermore, if π is a concrete system or process, then π is an actual dynamical phenomenon.

whose only components are position and *velocity*⁴¹.

⁴¹ If the mass is a constant m , and P_1 is a dynamical model of π which only contains position $Y_1[y \rho](t)$ and momentum $P_1[y \rho](t)$, then $\pi(P_1)$ is isomorphic to $\pi(P_2)$ whose components are position $Y_2[y v](t)$ and velocity $V_2[y v](t)$, where $Y_2[y v](t) = Y_1[y m v](t)$, and $V_2[y v](t) = P_1[y m v](t) / m$.

3. Frameworks and explanations

We have seen in section 2.3 that any dynamical study can be thought as the attempt to solve a specific dynamical problem $\langle \psi \ H \ F \rangle$, where ψ is a concrete system or process, H is the hypothesis " ψ is a dynamical phenomenon", and F is the request "find a specification of a dynamical model of ψ ". We now must consider more closely what a *specification* of a dynamical model is. It will then be clear that any such specification can be identified with a *possible* dynamical explanation, and that a *dynamical explanation of the concrete process ψ is just a solution of the dynamical problem $\langle \psi \ H \ F \rangle$ (that is, a specification of a dynamical model of ψ).*

3.1 Frameworks

Any dynamical model $P = \langle \langle M_j \dots M_k \ T \rangle \langle f_j \dots f_k \rangle \ O \rangle$ is a set theoretical entity. This abstract structure, however, can be linguistically *specified* by listing (i) all the magnitudes $M_j \dots M_k$ which occur in it; (ii) a definition " $f_j[x_j \dots x_k](t) = \alpha_j$ " ... " $f_k[x_j \dots x_k](t) = \alpha_k$ " for each evolution function in P ; (iii) the observable magnitudes in O . Condition (3c) of definition 2 ensures that any dynamical model can be specified in this way⁴². Very often, however, we do not directly list a definition

⁴² This condition ensures that for each evolution function $f_j[x_j \dots x_k](t)$ there is a function form α_j such that $f_j[x_j \dots x_k](t) = \alpha_j$. Therefore, each evolution function can be specified by listing the definition " $f_j[x_j \dots x_k](t) = \alpha_j$ ".

I use the word "specified" because, from the *epistemic* point of view, the definitions of the functions of a possible model P may not completely determine this model. In fact, the function forms in these definitions may contain constants whose values are unknown to us so that, by listing these definitions, we only *know* a *class* of possible models to which P belongs. From the *logical* point of view, however, each definition " $f_j[x_j \dots x_k](t) = \alpha_j$ " determines *the* function $f_j[x_j \dots x_k](t)$. Therefore, by listing all these definitions, the possible model P is linguistically determined, even

of each evolution function. What we do, instead, is specify a set of hypotheses A which entails a set of laws $\{ "f_j(t) = \alpha_j" \dots "f_k(t) = \alpha_k" \}$, and we then take the functions $f_j[x_j \dots x_k](t) \dots f_k[x_j \dots x_k](t)$ to be respectively defined by $\alpha_j \dots \alpha_k$. A possible dynamical explanation can thus be thought as having two components. First, a set of hypotheses A which entails a finite set of laws $\{ "f_j(t) = \alpha_j" \dots "f_k(t) = \alpha_k" \}$; second, a dynamical model $P = \langle \langle M_j \dots M_k \ T \rangle \langle f_j \dots f_k \rangle \ O \rangle$ whose evolution functions are defined by $\alpha_j \dots \alpha_k$. I will call this semi-linguistic structure a *framework*. A framework is minimal if its set of hypotheses A is identical to the set of laws $\{ "f_j(t) = \alpha_j" \dots "f_k(t) = \alpha_k" \}$. All this is expressed by the following definition:

Definition 6 (framework and minimal framework)

- (a) $\langle A \ P \rangle$ is a framework iff:
- (1) $P = \langle \langle M_j \dots M_k \ T \rangle \langle f_j \dots f_k \rangle \ O \rangle$ is a dynamical model;
 - (2) A is a set of hypotheses⁴³ which entails a set of laws $\{ "f_j(t) = \alpha_j" \dots "f_k(t) = \alpha_k" \}$, and $f_j[x_j \dots x_k](t) = \alpha_j, \dots$ and $f_k[x_j \dots x_k](t) = \alpha_k$.
- (b) $\langle A \ P \rangle$ is a minimal framework iff:
- (1) $P = \langle \langle M_j \dots M_k \ T \rangle \langle f_j \dots f_k \rangle \ O \rangle$ is a dynamical model;
 - (2) $A = \{ "f_j(t) = \alpha_j" \dots "f_k(t) = \alpha_k" \}$, and $f_j[x_j \dots x_k](t) = \alpha_j, \dots$ and $f_k[x_j \dots x_k](t) = \alpha_k$.

Below are four classic examples of possible explanations or frameworks.

though P may still remain partially unknown to us because function forms may contain constants whose values are unknown.

⁴³ For the moment, I do not impose any condition on the form of the hypotheses which are allowed to be members of A. Also, A may be redundant, in the sense that it may contain a proper subset A^* which also entails a set of laws $\{ "f_j(t) = \beta_j" \dots "f_k(t) = \beta_k" \}$ such that $f_j[x_j \dots x_k](t) = \beta_j, \dots$ and $f_k[x_j \dots x_k](t) = \beta_k$. Furthermore, the hypotheses in A may be ad hoc, in the sense that they may hold exclusively for a specific system or process ψ . These problems will be solved later, when I analyze the concept of a *theoretical framework*.

Example 6.1 (the Galilean framework for free fall)

In *Two New Sciences*, Galileo implicitly presupposes the definitions of velocity and acceleration (1914, ch. "Third day"). These two theoretical principles are expressed today by means of the derivative operator⁴⁴:

- [H₁] $\mathbf{V}(t) = D_t(\mathbf{Y}(t))$ (the velocity is the instantaneous rate of increase of the position)
[H₂] $\mathbf{A}(t) = D_t(\mathbf{V}(t))$ (the acceleration is the instantaneous rate of increase of the velocity)

When Galileo discusses the accelerated motion of a falling body (1914, 169) he explicitly assumes the specific hypothesis:

- [H₃] $\mathbf{A}(t) = c$ (the acceleration is equal to a constant; I take $c < 0$)

Let $G = \{H_1, H_2, H_3\}$, and P be the dynamical model defined in example 2.1. Then, $\langle G, P \rangle$ is a framework. We need only verify that G entails the laws of the position and velocity of a falling body, which are respectively expressed by:

- [λ₁] $Y(t) = y + vt + 1/2 ct^2$
[λ₂] $V(t) = v + ct$

Obviously, λ₂ follows from H₃ and H₂, and from λ₂ and H₁ we obtain⁴⁵ λ₁, whence $\langle G, P \rangle$ is a framework; $\langle G, P \rangle$ will be called the Galilean framework for free fall.

Example 6.2 (the Impetus framework for free fall)

According to Aristotelian dynamics, the velocity (not the acceleration) of a body is proportional to the total force acting on it. We can express this theoretical principle as:

- [H₁] $\mathbf{V}(t) = \mathbf{F}(t) / \mathbf{R}(t)$

where $\mathbf{R}(t)$ is a scalar which represents the resistance of a body to move (in the sense of changing *place*, not velocity) under the action of the total force $\mathbf{F}(t)$.

We can take the usual definition of velocity to be a second theoretical principle of Aristotelian dynamics. That is, the velocity is the instantaneous rate of increase of the position:

⁴⁴ Symbols in **bold** are syntactic *schemas*.

⁴⁵ This deduction is the integration of the equation of a straight line. Even though this problem can be solved today by any high school senior, it was by no means an easy problem at Galileo's time. Galileo arrived at the correct solution by employing an ingenious geometrical construction.

$$[H_2] \quad V(t) = D_t(Y(t))$$

According to Impetus theory, when a body is released with a certain positive force, this force (*impetus*) does not instantaneously vanish but, rather, decreases or increases with time. The impetus of the body will increase if another positive force (*propulsive force*) is acting on the body. It will decrease if, instead, the propulsive force is negative. We can now apply this idea to generate some specific hypotheses for the phenomenon of free fall. In this case, it is natural to assume that the only propulsive force is the weight of the body whose direction is downwards. Furthermore, since the body is moving, it must have a certain impetus. I thus assume that, at any time, the only two forces acting on the body are the weight and the impetus:

$$[H_3] \quad F(t) = W(t) + I(t)$$

It is also natural to assume that the weight and the resistance of the body to move, are two constants which possibly depend on the body. I take the positive direction of the y-axis to be upwards, so that we obtain:

$$[H_4] \quad W(t) = w, \text{ where } w < 0$$

$$[H_5] \quad R(t) = r, \text{ where } r > 0$$

We now must determine a law for the impetus. Since the weight continuously acts on the body, it must continuously produce additional impetus. Furthermore, since the weight is constant, it will produce additional impetus proportionally to the time during which it acts on the body. To obtain the impetus at an arbitrary time, we must add this acquired impetus to the initial impetus, so that we can write:

$$[H_6] \quad I(t) = i + wt, \text{ where } i \text{ is the initial impetus}$$

Let $I = \{H_1, H_2, H_3, H_4, H_5, H_6\}$, and P be the dynamical model defined in example 2.1. Then, $\langle I, P \rangle$ is a framework. We need only verify that the two laws

$$[\lambda_1] \quad Y(t) = y + vt + 1/2 ct^2$$

$$[\lambda_2] \quad V(t) = v + ct$$

are derivable from I . Let us deduce λ_2 first.

From $H_6, H_5, H_4, H_3,$ and H_1 we obtain:

$$1. V(t) = (w + i + wt) / r = (w + i)/r + tw/r$$

from 1:

$$2. V(0) = (w + i)/r = v$$

from 2 and 1, and since w and r are constants:

$$3. V(t) = v + ct$$

finally, from 3 and H_2 we obtain:

$$4. Y(t) = y + vt + 1/2 ct^2$$

Therefore, $\langle I, P \rangle$ is a framework; $\langle I, P \rangle$ will be called the Impetus framework for free

fall⁴⁶.

Example 6.3 (the Newtonian framework for free fall)

Newton's second principle of dynamics states that the acceleration of a body is proportional to the total force acting on it, that is:

$$[H_1] \quad \mathbf{A}(t) = \mathbf{F}(t) / \mathbf{M}(t)$$

where $\mathbf{M}(t)$ is the mass of the body (that is, the resistance of the body to change *velocity* under the action of $\mathbf{F}(t)$).

The usual definitions of velocity and acceleration are:

$$[H_2] \quad \mathbf{V}(t) = D_t(\mathbf{Y}(t)) \text{ (the velocity is the instantaneous rate of increase of the position)}$$

$$[H_3] \quad \mathbf{A}(t) = D_t(\mathbf{V}(t)) \text{ (the acceleration is the instantaneous rate of increase of the velocity)}$$

If we now consider the free fall of a body, it is natural to assume that the only force acting on the body is the weight, and that both the weight and the mass are constant. We thus obtain three specific assumptions:

$$[H_4] \quad \mathbf{F}(t) = \mathbf{W}(t)$$

$$[H_5] \quad \mathbf{W}(t) = w, \text{ where } w < 0$$

$$[H_6] \quad \mathbf{M}(t) = m, \text{ where } m > 0$$

Let $N = \{H_1, H_2, H_3, H_4, H_5, H_6\}$, and P be the dynamical model defined in example 2.1. Then, $\langle N, P \rangle$ is a framework. I now verify that the two laws

$$[\lambda_1] \quad Y(t) = y + vt + 1/2 ct^2$$

$$[\lambda_2] \quad V(t) = v + ct$$

are derivable from N . I first derive λ_2 .

⁴⁶ Galileo was aware of the possibility of deducing the laws for the velocity and the position of a falling body from Aristotelian dynamics and Impetus theory. In fact, in *Two New Sciences* (1914, 165-6), Sagredo clearly outlines how the velocity law can be deduced from the assumption that the impetus linearly decreases. Galileo, however, quickly dismisses Sagredo's suggestion by pointing out that we do not need an explanation of acceleration to deduce the two laws (1914, 166-7). Obviously, the explanation to which Galileo refers is the identification of acceleration with the ratio w/r , which follows from the hypotheses of the Impetus framework and Galileo's definition of acceleration. Galileo's point was probably also motivated by his conviction that the Aristotelian principle which states the proportionality of velocity and force was false. Even though Galileo never arrived at the modern formulation of the principle of inertia, he clearly recognized the possibility of motion in the absence of any force. This assumption is inconsistent with the Aristotelian principle.

From H_1 , H_4 , H_5 and H_6 :

1. $A(t) = w/m$

from 1, since w and m are constants:

2. $A(t) = c$

from 2 and H_3 :

3. $V(t) = v + ct$

finally, from 3 and H_2 we obtain:

4. $Y(t) = y + vt + 1/2 ct^2$

Therefore, $\langle N P \rangle$ is a framework; $\langle N P \rangle$ will be called the Newtonian framework for free fall⁴⁷.

Example 6.4 (the Baconian framework for free fall)

Consider the free fall of a body and suppose that, by applying inductive methods, we arrived at the following empirical generalizations:

$[\lambda_1]$ $Y(t) = y + vt + 1/2 ct^2$

$[\lambda_2]$ $V(t) = v + ct$

Let $B = \{\lambda_1 \lambda_2\}$, and P be the dynamical model defined in example 2.1. Then, $\langle B P \rangle$ obviously is a *minimal* framework; $\langle B P \rangle$ will be called the Baconian framework⁴⁸ for free fall.

3.2 Dynamical explanations

Intuitively, a *dynamical explanation* of a system or process is a set of hypotheses which entails all the evolution laws of a dynamical model of *that process*. To put it in a different way, a possible dynamical explanation (or a

⁴⁷ This is just one of the possible explanations of free fall which we can construct by employing Newton's theoretical principles. If we also assume Newton's principle of universal gravitation, we obtain a different framework, which is no longer consistent with the Galilean one. In fact, according to this principle, the acceleration of a falling body is not constant, but varies inversely with the square of the distance between the body and the earth.

⁴⁸ Francis Bacon never formulated this framework. The name, however, is appropriate because it is in fact possible to discover the laws for the velocity and position of a falling body by means of purely inductive methods. This has been demonstrated by the family of computer programs known as Bacon (Langley et al. 1987, 83-4).

framework) is an explanation of a given system just in case its set of hypotheses specifies a model *of that system*. An explanation of a system is minimal if it is a minimal framework which specifies a model of that system. This is expressed by the following definition:

Definition 7 (explanation and minimal explanation of a system or process)

- (a) $\langle A P \rangle$ is a dynamical explanation of π iff:
 $\langle A P \rangle$ is a framework, and P is a dynamical model of π ⁴⁹.
- (b) $\langle A P \rangle$ is a minimal dynamical explanation of π iff:
 $\langle A P \rangle$ is a minimal framework, and P is a dynamical model of π .

Example 7.1 (four competing explanations of free fall)

Let ψ be the free fall of a body, and assume that P (the dynamical model determined by the Galilean laws -- see example 2.1) is a dynamical model *of* ψ . Then, the Galilean framework, the Impetus framework, the Newtonian framework, and the Baconian framework are four different explanations of ψ which, however, specify the *same* dynamical model P . Also notice that the Baconian framework is a *minimal* explanation of ψ , while the other three are not minimal.

The Impetus framework and the Newtonian framework are inconsistent, because the second principle of dynamics contradicts the Aristotelian principle that velocity is proportional to force⁵⁰. The Galilean framework, instead, is logically compatible with

⁴⁹ Notice that π has some dynamical explanation $\langle A P \rangle$ just in case π is a dynamical phenomenon. If $\langle A P \rangle$ is a dynamical explanation of π , then P is a model of π , whence π is a dynamical phenomenon. Conversely, if π is a dynamical phenomenon, π has some model P , and the definition of a dynamical model implies that there is a minimal framework $\langle \{f_i(t) = \alpha_i\} P \rangle$. Therefore, $\langle \{f_i(t) = \alpha_i\} P \rangle$ is an explanation of π q.e.d. It thus follows that, if ψ is a concrete process or system, then ψ can be dynamically explained just in case ψ is an actual dynamical phenomenon. This formal result partially justifies my programmatic slogan "*all scientific explanations are explanations of specific types of phenomena*" (see the introduction of this chapter).

⁵⁰ This is true under the natural translation rule "total Newtonian force" = "total Aristotelian force". However, if we identify the total Newtonian force with the *propulsive force* of Impetus theory, the Impetus framework and the Newtonian framework are consistent. The propulsive force in Impetus theory is equal to the total Aristotelian force minus the impetus. In the case of free fall, the only propulsive force is the weight. If we then also introduce the concept of *acceleration* $\mathbf{A}(t)$,

either the Impetus or the Newtonian framework, since both IUG and NUG are consistent. Finally, the Baconian framework is consistent with the three other frameworks, because B follows from either G, I, or N.

The four frameworks also differ on the admissible interpretations for the constant c . The Baconian framework provides no interpretation for c . For Galileo, acceleration is the only admissible interpretation. For the Newtonian, c is either acceleration or the ratio between weight and mass (that is, the ratio between the weight and the resistance of the falling body to change *velocity* under the action of the weight). Finally, for the Impetus theorist, c is the ratio w/r , where w is the weight, and r is the resistance of the falling body to move ("move" in the sense of changing *place*, not velocity) under the combined action of the weight and the impetus.

Let me explicitly say that definition 7 is a *minimal* requirement which any dynamical explanation should satisfy. This definition is not intended to distinguish between 'deep' and 'shallow' explanations. Suppose that ψ is a concrete system or process, and that P is a dynamical model of ψ . Also suppose that somebody whom we trust (perhaps God the Great Mathematician) told us one of the laws " $f_i(t) = \alpha_i$ " for each evolution function f_i which is contained in P . Then, we could *ipso facto* produce an explanation of ψ , namely the *minimal* framework $\langle \{f_i(t) = \alpha_i\} P \rangle$. If some of the laws in $\{f_i(t) = \alpha_i\}$ contained some constant, the framework $\langle \{f_i(t) = \alpha_i\} P \rangle$ would not provide us with any interpretation for them⁵¹, and one might object that, for this reason, $\langle \{f_i(t) = \alpha_i\} P \rangle$ would not 'really explain' the process ψ . My view, however, is that this framework is not a *theoretical* explanation, but certainly is a *minimal* explanation of ψ , for its set of

and the concept of *resistance to change velocity under the action of the propulsive force* (which corresponds to our concept of *inertial mass* $M(t)$), we obtain $A(t) = w/r = a = W(t) / M(t) = w / M(t)$, whence $M(t) = w/a = m$ (notice that this entails $r = m$ so that, *as far as free fall is concerned*, we may freely identify our inertial mass with the Aristotelian resistance to move).

⁵¹ See example 6.4 for a concrete case.

hypotheses is one of the weakest set of hypotheses which determines all the evolution functions of the model P.

Computational models of cognitive processes may provide concrete examples of minimal dynamical explanations⁵². For instance, it can be proved that a version of the EPAM program determines a *dynamical model*⁵³. This model is intended to explain how humans learn a list of non-sense syllables (rote learning)⁵⁴. Furthermore, since the program is known, we can also specify a *law* for each evolution function of this model. Therefore, EPAM can be identified with a *minimal dynamical framework* $\langle \{f_i(t) = \alpha_i\} P \rangle$. The model of this minimal framework has also been empirically confirmed (at least partially), but the debate about its adequacy is still open. If the model specified by EPAM (or by some improved version of this program) is finally accepted as a dynamical model of the process of rote learning, EPAM will become an example of a *minimal dynamical explanation* of a cognitive process.

⁵² The method used for producing these computational models is not inductive. It is rather a specific version of what is traditionally called "resolution method". The resolution method is known in cognitive science as "means-end analysis". An appropriate name for the specific analysis involved in producing computational models of cognitive processes is "top-down functional analysis". See section 1.1 of chapter 4 for a brief sketch of this method.

⁵³ This proof is quite long and involves some technicalities. I plan to add it to an enlarged version of this dissertation. EPAM is a program which learns a series of non-sense syllables, (Feigenbaum 1959, 1963).

⁵⁴ Roughly speaking, the only observable of this model is the series of responses of a subject in a standard experiment. In one form of the experiment, a certain list of pairs of non-sense syllables is selected. The subject is presented the first syllable of each pair and he must respond with the second syllable. After the subject's response, the correct syllable is presented, until all the pairs in the list are exhausted. Then, the same list is presented again until the subject learns all the correct associations. In a second form of the experiment, instead, the syllables are presented in a sequence, and the subject must respond with the next syllable in the sequence.

3.3 Explanations as solutions of dynamical problems, and the inductive method

The definition of a dynamical explanation can also be interpreted as a definition of *the set of solutions* of a dynamical problem $\langle \psi \ H \ F \rangle$. In fact, an arbitrary solution of this problem is *a specification of a dynamical model of ψ* , and we have seen that any such specification can be identified with *a dynamical explanation of ψ* . Therefore, we can now express F (the request of the problem) in a different form, which is: "find a dynamical explanation of ψ ". I have suggested in section 2.3 that an arbitrary dynamical study can be thought as the attempt to solve a specific dynamical problem $\langle \psi \ H \ F \rangle$. We now see that *an arbitrary dynamical study can also be thought as the attempt to find a dynamical explanation of a given system or process*⁵⁵.

Suppose now that we are trying to dynamically explain a concrete system ψ or, equivalently, that we are trying to solve the specific dynamical problem $\langle \psi \ H \ F \rangle$. Also suppose that, in the course of this process, we perform the

⁵⁵ During the defense of this dissertation, Noretta Koertge raised the objection that, according to my view of a dynamical explanation, everything could be taken as the object of a dynamical study. This is true. However, when we decide that a concrete process or system ψ *should* be dynamically explained, we believe that understanding the *dynamical properties of this specific ψ* is *interesting, or worthwhile*. In other words, it is true that a problem of the form $\langle \psi \ H \ F \rangle$ can be posed for any ψ , but the fact that it can be posed does not entail that, for any ψ , this is an *interesting* problem. I am not proposing, and not even suggesting, that *any* problem of this form should be taken seriously (in other words, not any ψ is a dynamical *explanandum*). It is true, however, that this leaves open an important question which philosophers of science have overlooked so far (but Noretta Koertge is the exception to this rule): how do we evaluate and choose scientific *problems*? It is clear that, if science is a form of problem solving (as I believe it is), then *problem evaluation* is a crucial component of the scientific enterprise. Noretta Koertge has been working on a general theory of problem evaluation, which is based on the distinction of several dimensions with respect to which we judge problems. I believe that we badly need this theory. However, I also believe that, to understand the evaluation of scientific problems, we must also understand their *specific structure*. Therefore, I do not see an opposition between Noretta Koertge's approach and mine. Rather, I see them as being complementary.

following steps: **(1)** we (implicitly) assume the hypothesis of the problem H ; **(2)** we determine an appropriate number of observable magnitudes $M_j \dots M_r$; **(3)** we add a number of other magnitudes $M_m \dots M_k$ so that we obtain the component magnitudes $\langle M_j \dots M_r M_m \dots M_k \rangle$; **(4)** for each component magnitude M_p , we *induce* (or *guess on the base of data*) a law " $f_i(t) = \alpha_i$ "⁵⁶; **(5)** for each component magnitude M_p , we define its evolution function $f_i[x_j \dots x_k](t)$ by means of the equation " $f_i[x_j \dots x_k](t) = \alpha_i$ "; **(6)** we check that the evolution functions $\langle f_j \dots f_r f_m \dots f_k \rangle$ generate a dynamical system, that is, that they are related in such a way to satisfy [a] $f_i[x_j \dots x_k](0) = x_i$, and [b] $f_i[x_j \dots x_k](t+w) = f_i[f_j[x_j \dots x_k](t) \dots f_k[x_j \dots x_k](t)](w)$; **(7)** we check that, for each non-observable component M_s , there is an observable M_j such that its evolution function f_j depends on the initial value of M_s ; **(8)** for each component magnitude M_p , we empirically confirm its law " $f_i(t) = \alpha_i$ "; **(9)** we finally accept the hypothesis " P is a dynamical model of ψ ". Then, we have *inductively solved the problem* $\langle \psi H F \rangle$, or we have inductively produced a *minimal explanation* of the system ψ , namely the *minimal framework* $\langle \{ "f_i(t) = \alpha_i" \} P \rangle$, where $P = \langle \langle M_j \dots M_r M_m \dots M_k T \rangle \langle f_j \dots f_k \rangle \{ M_j \dots M_r \} \rangle$.

If the *inductive method* outlined above were always feasible, we should not

⁵⁶ Notice that, even if the magnitude M_j is not observable, it is possible to produce an informed guess of its law " $f_i(t) = \alpha_i$ " on the base of *all* the available data (that is, data concerning all the observable magnitudes). To produce this guess we must keep in mind the constraints that (1) *the time evolution functions of all magnitudes* must generate a dynamical system, and that (2) for any non-observable magnitude there must be an observable whose time evolution function depends on it. In this sense, it is possible to *induce* (or *guess on the base of data*) a law of *each* magnitude M_j (observable or not).

worry about theories at all, except for aesthetical reasons⁵⁷. Unfortunately, the problem with this method is that it does not always work. Very often, we cannot induce (or even guess) all the dynamical laws involved in a concrete process because, if we lack a theoretical perspective, we may not be able to determine an appropriate set of component magnitudes⁵⁸. Furthermore, what we can usually observe are not the laws of these magnitudes but, rather, some lawlike aspects which depend on these laws. These *lawlike generalizations*⁵⁹ may be very

⁵⁷ I would guess that, even in this case, we would still worry about theories, for aesthetical reasons are powerful ones. The point is that the inductive method can only provide us with *minimal* explanations, and we are not usually satisfied with this type of explanations. We feel that they should be subsumed under a more general point of view, which allows us to *synthesize* what we know in a few, simple, words. This systematic function is the characteristic feature of *theoretical* principles. But, perhaps, these aesthetical reasons are only the reflection of the great *pragmatic value* which theories have for us (for they are *efficient* tools for solving dynamical problems). If we had *powerful* inductive methods, our aesthetical judgements might very well change and, in that case, we might not care about theories so much.

⁵⁸ In the absence of a theoretical perspective, steps **(2)** and **(3)** of the inductive method may be extremely difficult. These two steps consist in determining: **(2)** the observables of a model; **(3)** other magnitudes on which the observable magnitudes depend. Even when all the magnitudes of a dynamical model of a concrete process are observable, it is not a trivial matter to realize that they are *sufficient* to specify a dynamical model of that concrete process. For example, it took the genius of Galileo to realize that position and velocity are sufficient to completely describe the dynamical behavior of a falling body. One of the great advantages of having a theory is that the theory may give us powerful heuristic rules for dealing with steps **(2)** and **(3)**. See sections 4.4 and 5.6.

⁵⁹ A typical example of this kind of generalizations are Kepler's three laws of planetary motion. These are not the laws which specify a dynamical model of planetary motion but, rather, they are mathematical relations which hold between the observable components of one particular orbit of that model. In fact, it so happens that only one particular orbit of this model is actually realized, and we can thus only observe portions of this orbit (I am assuming here that planetary motion is a dynamical phenomenon). Let me also explicitly say that, in general, there are *no laws of a dynamical phenomenon*. If a concrete process ψ is a dynamical phenomenon, then it has a dynamical model P , and the *laws* which specify the evolution functions of this model are the dynamical laws *of the model* (or more, precisely, *of the dynamical system generated by the model*). However, we cannot *seriously* talk of the laws *of the phenomenon*, unless we *identify* the concrete process ψ with the dynamical system $\psi(P)$ generated by P . This identification is justified as far as the dynamical properties of $\psi(P)$ are concerned but, beyond that, we should keep in mind that $\psi(P)$ and ψ are *different* systems. In fact, the property of being a law of $\psi(P)$ is not a dynamical property, for it is always possible to find a system which is isomorphic to $\psi(P)$ and does not have

important to later discover the dynamical laws, but they should not be confused with them. Induction very rarely, if ever, allows us to discover anything more than lawlike generalizations.

Fortunately, there is also a different approach which, in many cases, turns out to be more efficient. This is the *deductive method*, which consists in starting from natural *theoretical principles* and from plausible *specific assumptions*, and in then trying to deduce a set of laws which specify a *dynamical model*. If this theoretical work is successful, we end up with a *possible explanation* (or a framework), which is then provisionally accepted if all the deduced laws are empirically confirmed, and if the theoretical principles and the other assumptions cannot be criticized in any other way⁶⁰.

components.

⁶⁰ The acceptance of the hypothesis "<A P> is an *explanation* of ψ " involves two successive steps: **(1)** the acceptance of the hypothesis "P is an *empirically adequate* model of ψ "; **(2)** the acceptance of the hypothesis "P is a *dynamical model* of ψ ". The second step presupposes the previous one, but the justification methods involved in the first step are in general *different* from those involved in the second one. Furthermore, the acceptance of the hypothesis "<A P> is an *explanation* of ψ " should be distinguished from the acceptance of the stronger hypothesis "<A P> is an *adequate* explanation of ψ ". The acceptance of this stronger hypothesis also involves two successive steps: **(a)** the acceptance of the hypothesis "<A P> is an *explanation* of ψ "; **(b)** the acceptance of the hypothesis "A is an *adequate set of explanatory hypotheses* for ψ ". The justification methods for step **(b)** are *specific* to this step, and they may involve an evaluation of the *whole theory* to which the hypotheses in A belong.

4. Principles, laws, and theories

A dynamical theory is composed of *principles* and *laws*. Some of the principles are general, for they are supposed to hold for all the systems or processes which the theory intends to explain. They will be called *theoretical principles*. Other principles and all the laws are specific, that is, they are intended to hold only for one particular process or system. These principles and laws will be called *specific assumptions*. The theoretical principles determine a *set of frameworks* (or possible explanations). As a first approximation, we may thus identify a *dynamical theory* with a triple $\langle \Pi \ T \ \Phi \rangle$, where Π is a non-empty set of processes or systems (the intended *domain* of the theory), T is a non-empty set of *theoretical principles*, and Φ is a set of *frameworks* determined by T . In order to understand in which sense the theoretical principles determine the set of frameworks Φ , we need to look first at the nature of theoretical principles, specific principles, and laws.

4.1 Theoretical principles, specific principles, and laws

Intuitively, a *principle* is an equation which expresses the time evolution function of a magnitude as a mathematical function of the time evolution functions of other magnitudes. As a typical example, think of the second principle of dynamics, which is usually written as " $A(t) = F(t) / M(t)$ ".

Some principles are intended to *hold for all the time evolution functions of a specified type*. For example, the second principle of dynamics is valid for the time evolution functions of the acceleration, force, and mass of an *arbitrary* body. A

principle which satisfies this requirement is called a *theoretical principle*. It is thus clear that the *non-mathematical function constants which occur in a theoretical principle can be regarded as schemas which specify the type of evolution functions for which the principle is intended to hold*⁶¹. I will make explicit this fact by writing a *schema* in **bold**. For example, I will express the second principle of dynamics by means of the *equation schema* "**A(t) = F(t) / M(t)**". *A principle is theoretical if any non-mathematical symbol which occurs in it is a schema*⁶².

Besides theoretical principles, some explanations also employ other principles whose characteristic feature is that they are intended to *hold only for the system or process under investigation*. I call a principle of this kind a *specific principle*. For example, if we attempt to explain the free fall of a body, we may assume that the total force on the falling body is equal to its weight. This assumption is a principle, for it states a mathematical relation between the time evolution functions of two magnitudes. Nevertheless, the validity of this principle is limited to the particular process under investigation, namely free fall. *A principle is specific if no schema occurs in it*. For example, the specific principle mentioned above is expressed by the *equation* " $F(t) = W(t)$ ", where $F(t)$ and $W(t)$ are, respectively, the time evolution functions of the total force and of the weight of a falling body.

⁶¹ The traditional view takes a different approach. Theoretical principles are usually thought as universally quantified formulas. For example, the second principle of dynamics would be expressed by the formula: "for any x , $A(x, t) = F(x, t) / M(x, t)$ ". My point is that we do not need to quantify on objects in order to express theoretical principles. All what we need are *function schemas* " f_i " which specify *types* of time evolution functions. For example, if f_i is the time evolution function of the position of a *specific* body, the schema " f_i " expresses the time evolution function of the position of an *arbitrary* body.

⁶² Except the variable " t " whose range are the possible values of the magnitude time.

The third type of assumptions which are usually employed in theoretical explanations are *equations which specify the mathematical form of the time evolution function of a magnitude*. An equation of this kind is called a *law*. For example, to explain free fall, we may assume that the weight of the falling body is equal to a constant. We can express this law by means of the *equation* " $W(t) = c$ ". Laws, like specific principles, are intended to hold for a *fixed* system. Therefore, *no schema occurs in any law*. Unlike specific principles, however, *the right hand side of a law never contains a non-mathematical function term*.

In order to express principles and laws we need *individual variables* which vary on the possible values of magnitudes, and *function constants* which express the time evolution functions of magnitudes. I will use the following conventions. First, the variable " t " ranges on the set T of all possible values of the magnitude time. Second, if $M_i = \langle M_i \rangle$ is an arbitrary magnitude different from time, the variable " x_i " varies on the possible values of M_i , that is, on M_i . Third, when the function constant " f_i " occurs in the function term " $f_i(t)$ ", " f_i " expresses a function whose domain is T and whose codomain is the quantity M_i of the magnitude M_i . This function is called "the time evolution function of M_i ", and I will refer to this function by writing the function term " $f_i(t)$ " in *italics*. I also assume that we have specified *mathematical operators*⁶³ and *formation rules* which allow us to define the set of all *admissible function forms* $\{\alpha\}$, in such a way that no function constant " f_i " or

⁶³ For example, if the theory is classical mechanics, we will include all the usual algebraic and analytic operators.

function term " $f_i(t)$ " occurs in any function form⁶⁴. Let α be an arbitrary function form. I then define:

Definition 8 (principles and laws)

- (a) H is a theoretical principle iff:
H is an *equation schema* of the form " $f_i(t) = \alpha(f_j(t) \dots f_k(t) \mathbf{x}_s \dots \mathbf{x}_v t)$ ", where all the variable schemas " \mathbf{x}_s " ... " \mathbf{x}_v " and the variable "t" may be missing, but *at least one function term schema occurs in the right hand side*.
- (b) H is a specific principle iff:
H is an *equation* of the form " $f_i(t) = \alpha(f_j(t) \dots f_k(t) x_s \dots x_v t)$ ", where all the variables " x_s " ... " x_v " "t" may be missing, but *at least one function term occurs in the right hand side*.
- (c) H is a law iff:
H is an *equation* of the form " $f_i(t) = \alpha(x_s \dots x_v t)$ ", where all the variables " x_s " ... " x_v " "t" may be missing.

There is an important difference between the form of a principle and that of a law. A law does not contain any non-mathematical function term in its right hand side. Therefore, a law completely determines the function on the left side. A principle, instead, asserts that the functions on the right side must be in a specified mathematical relation to the function on the left side. Therefore, a principle only partially determines this function. The neat point is that this partial determination is possible even when all the functions related by the principle are *unknown*. In fact, to obtain a principle, *we need only know a mathematical relation between these functions*. This is one of the reasons why principles are so useful, for

⁶⁴ And, furthermore, for any function form α , and any function term " $f_i(t)$ ", " $f_i(t) = \alpha$ " must be a syntactically correct definition of " $f_i(t)$ " (where all the variables different from "t" which may occur in α are taken to be constants).

hypotheses about mathematical relations between unknown functions are in general easier to formulate than hypotheses which completely determine a function.

A second feature which makes principles extremely powerful is that *we can often deduce laws from theoretical principles and specific principles alone*. This is what we routinely do when the specific principles are differential equations⁶⁵. Suppose we have a theoretical principle which states that the time evolution of a magnitude of a first type Y is the derivative of the time evolution of a magnitude of a second type X . This can be expressed by:

$$[T] \quad Y(t) = D_t(X(t)), \text{ where "D}_t\text{" is the derivative operator.}$$

Suppose now that we want to find out the time evolution of the magnitude X for a specific system or process ψ . If we consider a *qualitative description of ψ* , we may be able to write a plausible mathematical relation between Y and X . This relation has the form of a specific principle, and it can be written as:

$$[S] \quad Y(t) = \alpha(X(t))$$

Since, by [T], $Y(t)$ is the derivative of $X(t)$, the specific principle [S] is a differential equation of the first order. We can now apply our mathematical knowledge and solve this equation. The result is a *law* of the form:

$$[L] \quad X(t) = \beta(x \ t) \quad \text{where } x \text{ is the initial value of the magnitude } X;$$

we have thus *deduced a law from principles alone*.

⁶⁵ I believe that there must be other ways of deducing laws from principles alone, but I cannot think of any other example.

4.2 Theoretical frameworks and theoretical explanations

I have proposed above to think of a dynamical theory as a triple $\langle \Pi \ T \ \Phi \rangle$, where Π is the intended *domain*, T is a non-empty set of *theoretical principles*, and Φ is a set of frameworks. I will call any framework $\langle A \ P \rangle \in \Phi$ a "*theoretical framework*". Theoretical frameworks have three characteristics. First, *each hypothesis in A is necessary for specifying the possible model P* . Second, *the set of hypotheses of a theoretical framework contains at least one of the theoretical principles of the theory*. Third, *all the other hypotheses are either specific principles or laws*. These three conditions are expressed by the following definition:

Definition 9 (theoretical framework and theoretical explanation)

- (a) $\langle A \ P \rangle$ is a theoretical framework relative to T iff:
 T is a non-empty set of theoretical principles, $\langle A \ P \rangle$ is a framework, and
- (1) let $P = \langle \langle M_j \dots M_k \rangle \langle f_j \dots f_k \rangle O \rangle$; then, for any $A^\# \subset A$, $A^\#$ does not entail⁶⁶ any set of evolution laws $\{ "f_j(t) = \beta_j" \dots "f_k(t) = \beta_k" \}$ such that $f_j[x_j \dots x_k](t) = \beta_j$,
... and $f_k[x_j \dots x_k](t) = \beta_k$;
 - (2) $A \cap T \neq \emptyset$;
 - (3) for any $H \in A$, if $H \notin T$, then H is either a specific principle or a law.

⁶⁶ In general, this condition is the most difficult to verify because, in order to prove that this condition holds we need to formalize the language in which the hypotheses are expressed. A condition which is easier to verify is the following: for any $H \in A$, H is used in a deduction of at least one of the laws $\{ "f_j(t) = \alpha_j" \dots "f_k(t) = \alpha_k" \}$, where $f_j[x_j \dots x_k](t) = \alpha_j$, ... and $f_k[x_j \dots x_k](t) = \alpha_k$. I do not attempt to formally define this condition, but I take that, *in each specific case*, we are able to distinguish deductions in which a hypothesis is used from those in which a hypothesis is redundant or irrelevant. For example, a deduction which first deduces β , then introduces the hypothesis γ , and then deduces β again under the hypothesis γ (by invoking the *a fortiori* law " $\beta \rightarrow (\gamma \rightarrow \beta)$ ") does not count as a deduction of β from γ where γ is used. The concept of relevant deduction can be made precise by using relevance logic. However, this presupposes that we formalize the language in which hypotheses are expressed. But then, if we must formalize, we can simply stick to standard logic and use the condition which I have stated.

- (b) $\langle A P \rangle$ is a theoretical framework iff:
 there is T such that $\langle A P \rangle$ is a theoretical framework relative to T .
- (c) $\langle A P \rangle$ is a theoretical explanation of π iff:
 $\langle A P \rangle$ is a theoretical framework, and P is a dynamical model of π .

Example 9.1 (theoretical and non-theoretical frameworks for free fall)

The Baconian framework for free fall (example 6.4) is not theoretical, for it does not satisfy condition (a2) of definition 9. The Galilean framework (example 6.1), the Impetus framework (example 6.2), and the Newtonian framework (example 6.3), instead, are theoretical⁶⁷.

If $\langle A^{\#} P \rangle$ is a framework, and $A^{\#} \subseteq A$, then I call $\langle A P \rangle$ an *extension* of $\langle A^{\#} P \rangle$. The definition of a theoretical framework implies that *the set of all theoretical frameworks and the set of all extensions of minimal frameworks are disjoint*. I take this formal result to express the sense of depth which we usually associate with theoretical explanations. This sense of depth depends on the fact that a set of laws which determines a dynamical model P is not given directly but, instead, is deduced from a set of hypotheses which contains at least one theoretical principle which is necessary for such deduction. The fact that no extension of a minimal framework is theoretical deserves to be stated as a theorem:

Theorem 2 (no extension of a minimal framework is theoretical)

If $\langle A^{\#} P \rangle$ is a minimal framework, and $A^{\#} \subseteq A$, then $\langle A P \rangle$ is not a theoretical framework.

⁶⁷ If condition (a1) of definition 9 is satisfied. However, notice that each hypothesis of each of these frameworks is *used* in the deduction of the Galilean laws. According to this intuitive criterion, then, we can take condition (a1) of definition 9 to be satisfied.

proof:

if $A^\# = A$, then, for any T , $\langle A P \rangle$ does not satisfy condition (a2) of definition 9. If $A^\# \subset A$, then, for any T , $\langle A P \rangle$ does not satisfy condition (a1) of definition 9 q.e.d.

4.3 Dynamical theories

I am now going to give a general definition of a *dynamical theory*. I have proposed above to think of a dynamical theory as having three components: an *intended domain* Π , a *set of theoretical principles* T , and a *set of frameworks* Φ . First, let me add the requirement that Φ be the set of all frameworks which are *theoretical with respect to* T . Second, let me complete this proposal by adding two other components.

We have seen before that, in order to express principles and laws, we must specify a set $\{\alpha\}$ of all the *admissible function forms*. This set may vary from theory to theory, so that $\{\alpha\}$ should be considered as a fourth component of each specific theory. This set can be regarded as the mathematical part of the theory. In general, the mathematics which a theory presupposes is not explicitly stated but, quite obviously, it is essential for precisely formulating the theoretical principles, the specific assumptions, and for deducing the laws which specify possible models of the systems or processes which the theory intends to explain.

The fifth component of a dynamical theory is associated with an interesting property which some of these theories satisfy. I have mentioned at the end of section 2.3 that classical mechanics can be taken to define a mechanical system as a system which has a dynamical model whose only components are position

and momentum. This definition can be interpreted as saying two different things. First, it specifies a particular class of dynamical models, that is, all those models whose only components are positions and momenta. This special set of models can be regarded as the set of all *standard models* of the theory. Second, it says that a system is in the intended domain (that is, a system is a *mechanical system*) just in case it has a standard model. This example thus suggests that a dynamical theory also has a fifth component. This fifth component is *a set of dynamical models* Σ which allows us to state a necessary and sufficient condition for a system π to be a member of the intended domain Π of the theory. This condition affirms that $\pi \in \Pi$ just in case there is $P \in \Sigma$ such that P is a dynamical model of π . I finally sum up all these observations with the following definition:

Definition 10 (dynamical theory)

$\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ is a dynamical theory iff:

- (1) Π is a non-empty set; (Π is the domain of the theory)
- (2) Σ is a set of dynamical models and, if $\Sigma \neq \emptyset$, then $\pi \in \Pi$ iff there is $P \in \Sigma$ such that P is a dynamical model of π ; (Σ is the set of standard models of the theory)
- (3) $\{ \alpha \}$ is a non-empty set of function forms; ($\{ \alpha \}$ is the set of admissible function forms of the theory)
- (4) T is a non-empty set of theoretical principles; (T is the set of theoretical principles of the theory)
- (5) Φ is a set of frameworks, and $\langle A P \rangle \in \Phi$ iff $\langle A P \rangle$ is a theoretical framework relative to T ; (Φ is the set of theoretical frameworks of the theory)

We thus see that a dynamical theory is a complex object. This object can be thought as having three parts: a non-linguistic part (the domain Π and the standard models Σ), a linguistic part (the mathematics $\{ \alpha \}$ and the theoretical principles T)

and, finally, a semi-linguistic part (the set of theoretical frameworks Φ) which connects the linguistic and non-linguistic parts. Traditionally, a theory has been conceived as either a set of sentences (by the *syntactic view*), or as a set of models (by the *semantic view*). Either view thus highlights an essential ingredient of any dynamical theory. Where both views go wrong, however, is in claiming that theories can be *reduced* to either a set of sentences or to a set of models.

The proponents of the semantic view have done an excellent job in pointing out the extremely important role of *models*. However, models have been thought in the *semantic sense* of structures which satisfy the theoretical principles. According to this view, *all models of a theory are semantic models*, in the specific sense that *all the theoretical principles are true in any model*. For example, the principles of classical mechanics define the predicate "is a classic particle system" and the models of the theory are all those structures which satisfy this predicate. It thus follows that all the principles of classical mechanics are true in any of its models.

In the first place, this view overlooks the important fact that *some models of a theory are models in a sense which is not semantic*. Rather, they are models in the same sense as *scale models* are, for they are *models of processes or systems*⁶⁸, and this means that a *similarity of structure* holds between a model and one of these systems. Giere has pointed out that scale models and theoretical models are alike, and he has also suggested that a model is *true of a certain system (or corresponds to that system)* if there is a similarity of structure

⁶⁸ A model of a theory may be a model in this sense even though it is not a model in the semantic sense of satisfying the theoretical principles. See below for concrete examples.

between the model and this system (1984, 1985, 1988). Van Fraassen has proposed that this similarity of structure should be thought as an *isomorphism* (1989, 226). However, as far as I know, these informal remarks have not been spelled out in detail for any specific theory. My view is that, for dynamical theories, the similarity of structure between a dynamical model P and a concrete system ψ is an isomorphism between *dynamical systems*, that is, an isomorphism between the dynamical system $\psi(P)$ generated by the model and the concrete system ψ . If this isomorphism holds, then the dynamical system generated by the model and the concrete system have the same *dynamical properties*, and the same *modal structure*. I believe that, as far as dynamical theories are concerned, this concept formally expresses the so called "correspondence theory of truth". However, it should be noticed that *the concept of dynamical model of a concrete system is not a semantic concept*, for the isomorphism between the dynamical system generated by the model and the concrete system has nothing to do with the satisfaction relation between the theoretical principles and the model.

In the second place, this formal version of the idea of truth as correspondence should not be confused with Tarski's analysis of truth for formal languages. The only connection with Tarski's theory is a trivial one, namely, that the hypothesis "P is a dynamical model of ψ " is true just in case P is a dynamical model of ψ . Obviously, this does not explain what "P is a dynamical model of ψ " means. This point needs to be stressed, for it is usually misunderstood. The so called "correspondence theory of truth" does *not* affirm that a *sentence* corresponds to

a *fact*. Rather, it affirms that a *model* of a certain *type* (often an abstract model) corresponds to a concrete *system*⁶⁹. Therefore, "P corresponds to ψ " cannot be analyzed as: " ψ " is true iff ψ ⁷⁰. The correct T-sentence for "P corresponds to ψ " is: "P corresponds to ψ " is true iff P corresponds to ψ . It is thus clear that T-sentences do not provide us with any explication of the idea of truth as correspondence. In general, I believe that this concept must be analyzed in terms of an *isomorphism* which holds between a *structure* $\psi(P)$ *generated by the model* P and the *system* ψ . Therefore, we cannot specify *one isomorphism* which formally expresses *the* theory of truth as correspondence, for this isomorphism depends on the *type of structure* which $\psi(P)$ is. Furthermore, the *type of model* which P is also depends on the *type of structure* which $\psi(P)$ is. In other words, there cannot be *only one formal theory* of truth as correspondence. Rather, there will be *many formal theories*, which depend on *the type of structure* involved in such correspondence. What I have attempted to do is to formulate *one* of these formal theories, namely, the formal theory of truth as correspondence when the

⁶⁹ What I mean is that the concept of truth as correspondence, *as typically used in empirical science*, does not affirm that a *sentence* corresponds to a *fact*. Obviously, there is an intuitive concept of truth as correspondence which affirms just this. *This* concept has been explicated by Tarski, and we all know that Tarski's theory of truth is relevant to logic and meta-mathematics. My point is that truth as correspondence between sentences and facts is largely irrelevant to empirical science. Instead, the appropriate concept in this domain is truth as correspondence between *models* and *systems*.

⁷⁰ If ψ were a fact, then ψ could be expressed by a sentence " ψ " and thus, by taking $P = "$ ψ ", the T-sentence for "P corresponds to ψ " would be: " ψ " is true iff ψ .

structures involved are *dynamical systems*⁷¹.

In the third place, the characteristic function of theoretical principles is not that of defining a set of *models which satisfy the theoretical principles*. Rather, theoretical principles define a set of *theoretical frameworks*, and the hypotheses of each of these frameworks specify one model. Ultimately, theoretical principles thus determine a set of models. However, *the models in this set do not usually satisfy the theoretical principles*. In other words, *it is false that all models of a theory are models in the semantic sense of satisfying the theoretical principles*, for some dynamical theories determine models in which some theoretical principles are neither true nor false. Therefore, the semantic view is contradicted by this property of dynamical theories. As a typical example think of the model specified by the Newtonian framework for free fall (example 6.3). Even though the second principle of dynamics is used to specify this model, this principle is not true in this model, for mass, force, and acceleration are not components of this model⁷².

We must now consider more closely the roles which the set of *standard* models

⁷¹ My *general* proposal for analyzing "P corresponds to ψ " is this. "P corresponds to ψ " should be analyzed as "P is a Δ -model of ψ ", where Δ is a *specified type of structure*. "P is a Δ -model of ψ " should then be defined as "P is a Δ -model, and $\psi(P)$ is Δ -isomorphic to ψ ", where the Δ -isomorphism is the isomorphism appropriate for the type of structure Δ , and a Δ -model is defined in such a way that each Δ -model P *generates*, or *induces*, exactly one structure $\psi(P) \in \Delta$.

⁷² I am *not* saying that the second principle of dynamics is *false* in the Galilean model of free fall. I am just saying that, in this model, the second principle of dynamics is neither true nor false, just because this model does not provide any interpretation for the function terms "A(t)", "F(t)", and "M(t)". In fact, the only components of the model are position and velocity. This example is typical. Even the standard models of classical mechanics do not satisfy the theoretical principles, for the only components of these models are positions and momenta.

Σ plays in a theory. In the first place, if this set is not empty, we are given a necessary and sufficient condition for a system or process π to be a member of the intended domain Π : $\pi \in \Pi$ just in case there is a model $P \in \Sigma$ such that P is a dynamical model of π . Notice that this condition implies that *any system π in the intended domain is a dynamical phenomenon. Furthermore, if π is a concrete system or process, then π is an actual dynamical phenomenon.* We thus see that any dynamical theory whose set of standard models is not empty determines a specific subset of the set of all dynamical phenomena. The dynamical phenomena in this subset are exactly the processes or systems in the intended domain Π . We will see later that this fact has an important consequence with respect to the *methods* which we may use to solve a concrete dynamical problem $\langle \psi \ H \ F \rangle$ ⁷³.

In the second place, the set of standard models Σ is sometimes given in such a way that each standard model turns out to be *completable* with respect to the magnitudes determined by the theoretical principles. Let me explain with an example. Consider classical mechanics, where the set of standard models Σ is the set of all dynamical models whose components are the positions and momenta of n bodies ($n > 0$), and the set of theoretical principles T includes the principle of universal gravitation, the second principle of dynamics, and the definitions of velocity, acceleration, and momentum⁷⁴. For each standard model $P \in \Sigma$, these

⁷³ Recall that $\langle \psi \ H \ F \rangle$ is a dynamical problem just in case ψ is a concrete system or process, H is the hypothesis " ψ is a dynamical phenomenon", and F is the request "find a dynamical explanation of ψ ".

⁷⁴ These five theoretical principles are respectively expressed by the following schemas:
 $[T_1] \mathbf{G}_{ab}(t) = (k \mathbf{M}_a(t) \mathbf{M}_b(t) / |\mathbf{Y}_b(t) - \mathbf{Y}_a(t)|^3) (\mathbf{Y}_b(t) - \mathbf{Y}_a(t))$
 where $k > 0$, $\mathbf{G}_{ab}(t)$ is the gravitational force on body \mathbf{a} exerted by body \mathbf{b} , $\mathbf{Y}_a(t)$ is the position

principles determine a set of magnitudes $T(P)$ whose members are expressed by the schemas: G_{ij} = the gravitational force on body i exerted by body j ; F_i = the total force on body i ; M_i = the mass of body i ; Y_i = the position of body i ; V_i = the velocity of body i ; A_i = the acceleration of body i ; P_i = the momentum of body i . Any standard model $P \in \Sigma$ includes the positions and momenta of n bodies ($n > 0$). Therefore, $P = \langle \langle Y_1, P_1 \dots Y_n, P_n \rangle T \rangle \langle Y_1(t), P_1(t) \dots Y_n(t), P_n(t) \rangle O$, where $O = \{Y_1 \dots Y_n, T\}$. Now, even though the magnitudes G_{ij} , F_i , M_i , V_i and A_i are not components of P , a set of laws of these magnitudes is derivable from the theoretical principles and from a set of laws of the magnitudes in P ⁷⁵. In this sense, each standard model of classical mechanics is *completable* with respect to the magnitudes determined by the theoretical principles. In general, a theory is *complete* just in case each of its standard models is completable. Therefore, we have just verified that *classical mechanics is complete*⁷⁶. The following definition

of \mathbf{a} , $M_a(t)$ is the mass of \mathbf{a} , $Y_b(t)$ is the position of \mathbf{b} , and $M_b(t)$ is the mass of \mathbf{b} . Position and gravitational force are vectors and, if v is a vector, $|v|$ is its modulus. If $\mathbf{a} = \mathbf{b}$, then $|Y_b(t) - Y_a(t)| = 0$, so that $G_{ab}(t)$ is not determined by $[T_1]$. I set $G_{ab}(t) = 0$ for convenience.

[T₂] $\mathbf{A}(t) = \mathbf{F}(t) / \mathbf{M}(t)$

where $\mathbf{A}(t)$ is the acceleration, $\mathbf{M}(t) > 0$ is the mass, and $\mathbf{F}(t)$ is the total force acting on an arbitrary body;

[T₃] $\mathbf{V}(t) = D_t(Y(t))$

where $\mathbf{V}(t)$ is the velocity, $Y(t)$ the position of an arbitrary body, and " D_t " is derivative operator;

[T₄] $\mathbf{A}(t) = D_t(\mathbf{V}(t))$

[T₅] $\mathbf{P}(t) = \mathbf{M}(t)\mathbf{V}(t)$ where $\mathbf{P}(t)$ is the momentum of an arbitrary body.

⁷⁵ If $Y_i(t) = \alpha_i$ is given, then, by the definition of velocity, we deduce $V_i(t) = D_t(\alpha_i)$. From this and the definition of momentum, since $P_i(t) = \beta_i$ is given, we deduce $M_i(t) = \beta_i / D_t(\alpha_i)$. From the definition of acceleration, $A_i(t) = D_t(D_t(\alpha_i))$. From the second principle of dynamics, $F(t) = ((\beta_i / D_t(\alpha_i)) D_t(D_t(\alpha_i)))$. Finally, from the principle of universal gravitation, $G_{ij}(t) = (k (\alpha_j - \alpha_i) (\beta_j / D_t(\alpha_j)) (\beta_i / D_t(\alpha_i))) / |\alpha_j - \alpha_i|^3$.

⁷⁶ The third principle of dynamics is expressed by the schema:

[E] $\mathbf{F}_{ab}(t) = -\mathbf{F}_{ba}(t)$, where $\mathbf{F}_{ab}(t)$ is the force on body \mathbf{a} exerted by body \mathbf{b} .

If we take the *theoretical* principles of classical mechanics to also include this principle, then

expresses the concepts of completable model and complete theory:

Definition 11 (complete theory)

(a) P is completable with respect to T(P) iff:

$P = \langle \langle M_j \dots M_k \rangle \langle f_j \dots f_k \rangle O \rangle$ is a dynamical model, T(P) is the set of magnitudes determined by a set of theoretical principles T, and for any magnitude $M_r \in T(P)$, a law " $f_r(t) = \alpha_r$ " is derivable from T and from a set of laws {" $f_j(t) = \alpha_j$ " ... " $f_k(t) = \alpha_k$ "} such that $f_j[x_j \dots x_k](t) = \alpha_j$, ... and $f_k[x_j \dots x_k](t) = \alpha_k$.

(b) $\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ is complete iff:

$\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ is a dynamical theory, $\Sigma \neq \emptyset$ and, for any $P \in \Sigma$, P is completable with respect to T(P).

Theorem 3 (completeness of classical mechanics)

Classical mechanics is complete

proof:

see the foregoing discussion.

4.4 The deductive method for solving dynamical problems, and the heuristic value of dynamical theories

We have seen in the previous sections that any dynamical study can be thought as the attempt to solve a specific dynamical problem $\langle \psi H F \rangle$, where ψ is a concrete system or process which we want to explain, H is the hypothesis " ψ is a dynamical phenomenon", and F is the request "find a dynamical explanation of ψ ". I am now going to consider how we may try to solve this

classical dynamics is no longer complete. However, I believe that the third principle of dynamics is better interpreted as a *heuristic* principle which helps us to determine an appropriate set of *specific assumptions* for a particular system or process we want to explain.

problem if we are given a dynamical theory $\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ ⁷⁷.

Suppose then that we are trying to dynamically explain a concrete system ψ or, equivalently, that we are trying to solve the specific dynamical problem $\langle \psi H F \rangle$, and that we are given a dynamical theory $\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ such that $\Sigma \neq \emptyset$ ⁷⁸. Also suppose that, in the course of this process, we perform the following steps: **[1]** we assume the hypothesis " $\psi \in \Pi$ "⁷⁹; **[2]** *keeping in mind that* $\psi \in \Pi$ *just in case there is* $P \in \Sigma$ *such that* P *is a dynamical model of* ψ , we determine an appropriate number of observable magnitudes $M_j \dots M_r$ and **[3]** we add a number of other magnitudes $M_m \dots M_k$, so that we obtain the component magnitudes $\langle M_j \dots M_r M_m \dots M_k \rangle$ ⁸⁰; **[4]** we select an appropriate subset of the set of theoretical principles T ; **[5]** we add some specific principles or laws, so that we obtain a set of hypotheses A ⁸¹; **[6]** for each component magnitude M_p , we *deduce*

⁷⁷ A dynamical theory $\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ is *given* just in case Σ , $\{ \alpha \}$, and T are specified. The specification of Σ , $\{ \alpha \}$, or T need not be explicit. The ideal case is when: (i) Σ is explicitly defined; (ii) $\{ \alpha \}$ is defined by specifying admissible mathematical operators and formation rules; (iii) T is a finite set of theoretical principles which are explicitly listed.

⁷⁸ If $\Sigma = \emptyset$, then the first three steps below should be replaced by the corresponding steps of the *inductive method* (see section 3.3).

⁷⁹ Notice that, since $\Sigma \neq \emptyset$, the hypothesis " $\psi \in \Pi$ " implies the hypothesis of the problem " ψ is a dynamical phenomenon".

⁸⁰ For example, if the theory is classical mechanics, step **[1]** consists in assuming that ψ is a mechanical system, that is, that there is a model of ψ whose only components are the positions and momenta of n bodies. Steps **[2]** and **[3]** thus reduce to the following heuristic rules: **(a)** determine n bodies whose positions and momenta might be components of a standard model of ψ ; **(b)** choose components $\langle Y_i P_i \dots Y_n P_n T \rangle$, where Y_i and P_i are, respectively, the position and momentum of body i .

⁸¹ In the case of classical mechanics steps **[4]** and **[5]** often reduce to the following heuristic rules: **(a)** choose the second principle of dynamics, and the definitions of acceleration, velocity, and momentum; **(b)** for each body, write a law which states that its mass is a constant; **(c)** for each body which is part of the model, individuate all the forces acting on it, and write a specific principle

a law " $f_i(t) = \alpha_i$ " from the hypotheses A; [7] for any $H \in A$, we check that we have used H in the deduction of at least one of the laws {" $f_j(t) = \alpha_j$ " ... " $f_k(t) = \alpha_k$ "}; [8] for each component magnitude M_j , we define its evolution function $f_j[x_j \dots x_k](t)$ by means of the equation " $f_j[x_j \dots x_k](t) = \alpha_j$ "; [9] we check that the evolution functions $\langle f_j \dots f_r, f_m \dots f_k \rangle$ generate a dynamical system, that is, that they are related in such a way to satisfy [a] $f_j[x_j \dots x_k](0) = x_j$, and [b] $f_j[x_j \dots x_k](t+w) = f_j[f_j[x_j \dots x_k](t) \dots f_k[x_j \dots x_k](t)](w)$; [10] we check that, for each non-observable component M_s , there is an observable M_r such that its evolution function f_r depends on the initial value of M_s ; [11] for each component magnitude M_j , we empirically confirm its law " $f_j(t) = \alpha_j$ "; [12] we finally accept the hypothesis "P is a dynamical model of ψ ". Then, we have *deductively solved the problem* $\langle \psi, H, F \rangle$, or we have deductively produced a *theoretical explanation* of the system ψ , namely the *theoretical framework* $\langle A, P \rangle$, where $P = \langle \langle M_j \dots M_r, M_m \dots M_k \rangle \langle f_j \dots f_k \rangle, O \rangle$ and $O = \langle M_j \dots M_r \rangle$.

I will call the method which I have just sketched the "*deductive method*". If we compare this method with the inductive one (see section 3.3), we see that steps [8]-[12] of the deductive method are the same as steps (5)-(9) of the inductive one. Furthermore, steps [1]-[3] correspond to steps (1)-(3), and step [6]

which states that the total force on that body is equal to the sum of all the forces acting on it. To simplify this task, keep in mind all the action-reaction pairs; (d) if some of these forces are gravitational, choose the principle of universal gravitation; (e) if some gravitational force is exerted by a body which is not part of the model, try to determine a simple law for its position; (f) for the forces which are not gravitational, try to determine a simple law first or, if this fails, a specific principle. To simplify this task, keep in mind all the action-reaction pairs. The right hand side of each specific principle should only contain the positions and/or velocities of bodies which are parts of the model.

corresponds to step (4). The steps which do not have any inductive analogue are thus [4], [5], and [7]. These steps (together with [6] and [8]-[10]) ensure that the result of the deductive method, that is $\langle A, P \rangle$, is a *theoretical* framework. We have seen before that the result of the inductive method is instead a *minimal* framework.

The two methods also differ with respect to their *generality* and to their *power*. *The inductive method is more general than the deductive one, however, as a rule, the deductive method is more powerful.* The generality of the inductive method depends on step (1), for this step only presupposes that ψ is a dynamical phenomenon. Step [1] of the deductive method, instead, affirms that ψ belongs to a *subset* of the set of all dynamical phenomena, so that this method is more specific.

This decrease in generality, however, is usually compensated by an increased power. This is due to the fact that steps [2]-[3], [4]-[5], and [6] are usually associated with *standard heuristic rules* which depend on the specific theory $\langle \Pi, \Sigma, \{ \alpha \}, T, \Phi \rangle$. The heuristic rules which permit us to efficiently perform step [6] are logical and mathematical. The corresponding step of the inductive method (step (4)) is also associated with heuristic rules, but these rules are inductive. In general, inductive heuristic is less efficient than logical or mathematical heuristic. The heuristic rules associated with steps [4]-[5] depend on the formulation of the theoretical principles, and on the mathematics of the theory. If the theory is a good one, these rules may be extremely powerful. Finally, the rules associated with steps [2]-[3] depend on the definition of the set of standard models of the

theory. Again, depending on the theory, these rules may be very efficient. By contrast, there are no standard and powerful heuristic rules to perform the corresponding steps of the inductive method (steps **(2)** and **(3)**).

We thus see that *a dynamical theory may be an extremely useful tool for solving dynamical problems*. I believe that the *heuristic value* of theories is the primary factor to consider if we are trying to understand why theories are so important to us. Realists typically subscribe to a different view. They interpret theories as capable of being true or false, in the sense that

there are things in the world to which our laws and theories refer and of which they are true or false. They are, that is, to be understood as referring to real existents and ascribing genuine properties to them. (Ellis 1985)

It thus follows that realists primarily value theories because they would reveal to us *deeper levels of reality*, which go far beyond what we can apprehend in the course of our perceptual experience, or in our practical interactions with the external world. Even though I am a convinced realist, I maintain that *dynamical theories* do not have any especially deep ontological consequence.

First, let me explicitly say that *the correct realistic interpretation of dynamical theories does not imply that theoretical principles, specific principles, or laws have truth values*. This property would follow if we subscribed to a theory of truth as correspondence between *sentences* and *facts*. However, this is not the appropriate concept for dynamical theories. The correspondence relation does *not* hold between the *hypotheses* of the theory and the *facts* of the external world but, rather, between some *models* determined by the theory and some concrete

systems or processes. Therefore, the theoretical principles (or the specific assumptions) are not objectively true or false. Instead, objective truth or falsehood is a property of *the models* of the theory. To use a Popperian term, the *truth content* of a dynamical theory is the set of all its models which are models of some concrete system or process⁸². The hypotheses of the theory *specify* or *determine* these models, but these hypotheses are neither true nor false *of the systems* to which these models correspond⁸³.

In the second place, what a dynamical theory reveals to us is, at most, that some concrete process or system is a *specific dynamical phenomenon*. Even though this may be a very important discovery, by no means does it reveal a deep ontological level. Let me explain. If P is a model determined by a dynamical theory and, furthermore, P is a dynamical model of a concrete system ψ , then all what we know is that ψ is isomorphic to the dynamical system $\psi(P)$ generated by P. This fact has only two ontological consequences: **(a)** that ψ has *dynamical properties* which are the same as the dynamical properties of the system $\psi(P)$ and **(b)** that ψ has a quite simple *modal structure*, that is, the modal structure proper of any dynamical system. Therefore, the ontological implications of dynamical theories are quite minimal, and I take that they are not sufficient to explain why

⁸² The *content* of a theory $\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ is the set $C(T)$ of all dynamical models determined by the theory, that is, $C(T) =: \{ x: \text{there is } \langle A P \rangle \text{ such that } \langle A P \rangle \text{ is a theoretical framework relative to } T, \text{ and } P = x \}$. The *truth content* $C_T(T)$ is the subset of $C(T)$ which contains all those models which are models of some concrete system, that is, $C_T(T) =: \{ P: P \in C(T), \text{ there is } \psi \text{ such that } \psi \text{ is a concrete system or process, and } P \text{ is a dynamical model of } \psi \}$.

⁸³ Furthermore, we have also seen that, typically, the theoretical principles are neither true nor false *of the models*.

these theories are so valuable to us. *My view is that the most important component of the value of dynamical theories is not epistemological but rather pragmatic, in the specific sense that these theories may be powerful tools for producing dynamical explanations*⁸⁴.

The *heuristic value* of a dynamical theory $\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ depends on the power of three sets of rules which are typically associated with specific components of the theory: **(i)** the rules associated with the definition of the set of standard models Σ - these rules allow us to select appropriate sets of *component magnitudes*; **(ii)** the rules associated with the theoretical principles T and their mathematical formulation - these rules allow us to select appropriate sets of *hypotheses*; **(iii)** the rules associated with the logic and mathematics of the theory $\{ \alpha \}$ - these rules allow us to *deduce a law for each component magnitude*. We will see in the next section that the *heuristic value* of dynamical theories may also play an important role in the *choice* between two competing theories.

⁸⁴ Let $C(T)$ be the set of all dynamical models determined by a theory $\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$, that is, $C(T) = \{ x: \text{there is } \langle A P \rangle \text{ such that } \langle A P \rangle \text{ is a theoretical framework relative to } T, \text{ and } P = x \}$. Then, there is always a set of *minimal frameworks* which determines exactly the models in $C(T)$. Intuitively, $C(T)$ is the complete *content* of the theory. Therefore, there is nothing which a theory says of the world which cannot be said without a theory. What this means is that, if we only consider *the epistemological value* of theories, then they are on a par with sets of minimal frameworks. Therefore, what makes theories more valuable must be some other property which sets of minimal frameworks do not have. My view is that this property is the *heuristic value* of a theory.

5. Two examples of dynamical theories

I am now going to discuss two examples of dynamical theories. The first is a theory which is based on 'unorthodox' physical intuitions. This is a version of what is known as *Aristotelian dynamics*, augmented with some ideas taken from *Impetus dynamics*. The basic intuitions which lead to the formulation of the principles of this theory are extremely natural, and are familiar to any student of the history of science. The only further constraints which I have imposed to generate these principles are that (i) they be expressed in *current mathematical language*, and that (ii) they turn out to be *theoretical principles in the sense of definition 8*. I call this first theory "*Impetus theory*". The second theory is the 'purely dynamical' fragment of classical mechanics, that is, the theory whose principles are the definitions of velocity and acceleration and the second principle of dynamics⁸⁵. I call this theory "*Newtonian dynamics*". The surprising result, which I will prove in section 5.5, is that Newtonian dynamics turns out to be *inconsistent* with Impetus theory and *translatable* into Impetus theory. The converse is also true. The exact sense in which this is possible will be clear later. For the moment, let us see what Impetus theory says.

⁸⁵ There is really nothing 'more dynamical' in this theory than, say, in the fragment which only contains the definitions of velocity and acceleration. However, there is a *traditional* meaning of the term 'dynamics' as the study of motion with reference to the *forces* involved. According to this usage, then, dynamics starts with the second principle of dynamics, while the definitions of velocity and acceleration belong to the domain of *kinematics*.

5.1 The theoretical principles of Impetus theory (or Aristotelian & Impetus dynamics)

I take the usual definition of velocity to be the *first principle of Aristotelian dynamics*⁸⁶. This theoretical principle can be expressed by the equation-schema:

[A₁] $\mathbf{V}(t) = D_t(\mathbf{Y}(t))$, where $\mathbf{V}(t)$ is the velocity of an arbitrary body, $\mathbf{Y}(t)$ is its position, and " D_t " is the derivative operator.

The *second principle of Aristotelian dynamics* states that the velocity of a body is directly proportional to the total force acting on it, and inversely proportional to the intrinsic resistance of the body to move. This can be expressed by the equation-schema:

[A₂] $\mathbf{V}(t) = \mathbf{F}(t) / \mathbf{R}(t)$, where $\mathbf{R}(t) > 0$, $\mathbf{F}(t)$ is the total force, and $\mathbf{R}(t)$ is the resistance of an arbitrary body to move under the action of $\mathbf{F}(t)$.

The magnitude \mathbf{R} has some analogy with our concept of inertial mass. Notice, however, that the inertial mass is the resistance of a body to *change velocity*, while \mathbf{R} is the resistance of a body to *change position*. Also, the total force is velocity multiplied by resistance to move. It is thus somehow similar to our momentum. But, again, this must be taken *cum grano salis*, because the resistance to move is not the inertial mass.

According to Impetus theory, the total force acting on a body can always be divided into two components: the propulsive force, and the impetus. This principle can be expressed by:

⁸⁶ Obviously, this is not historically correct, for the concept of derivative is a modern one. Nevertheless, I am not interested here in historical accuracy. My goal is to use current mathematical language to state a set of theoretical principles which express the basic intuitions underlying Aristotelian and Impetus dynamics.

[I₁] $\mathbf{F}(t) = \mathbf{P}(t) + \mathbf{I}(t)$, where $\mathbf{I}(t)$ is the impetus, and $\mathbf{P}(t)$ are all the other forces acting on the body.

The impetus is *an internal force of the body whose change produces a change in its velocity*. The propulsive force, instead, is *the external agent which causes the body to change its impetus*. The ultimate effect of the propulsive force thus is an acceleration. In this sense, the propulsive force can be identified with the Newtonian total force. The way in which an acceleration is produced, however, is quite different. The propulsive force 'consumes' or 'produces' impetus, and it is this impetus change which then accounts for a change in velocity⁸⁷.

From [I₁] and [A₂] we obtain an equation-schema which allows us to express the impetus by means of $V(t)$, $R(t)$, and $P(t)$:

$$[1] \quad \mathbf{I}(t) = \mathbf{V}(t)\mathbf{R}(t) - \mathbf{P}(t)$$

From [1], if \mathbf{i} , \mathbf{v} , \mathbf{r} , and \mathbf{p} are, respectively, the initial impetus, velocity, resistance to move, and propulsive force, we obtain:

$$[2] \quad \mathbf{i} = \mathbf{vr} - \mathbf{p}$$

We thus see that the initial impetus has always two components. The first is equal to the initial velocity multiplied by the initial resistance to move. The second component is equal to the opposite of the propulsive force acting on the body at time 0. It is thus equal to the force which would be present if the body were at rest at time 0.

What about the impetus at any other time? The basic idea of Impetus theory

⁸⁷ The concept of impetus closely corresponds to what, in every day English, is called "momentum". The Italian word for the same folk concept is "impeto".

is this. If a body is released with an initial impetus i , this force does not instantaneously vanish but, rather, it either decreases or increases, depending on whether the propulsive force is negative or positive. To understand what this exactly means, suppose first that the propulsive force is constant, that is, $P(t) = p$ for any t . In this simple case, the impetus at an arbitrary time t is given by the equation:

$$[3] \quad I(t) = i + pt$$

The meaning of this equation is as follows. If p is positive, this force produces additional impetus, and this acquired impetus is proportional to the time during which the force p acts. If p is negative, p will instead produce an impetus loss which is proportional to the time during which the negative force acts. Finally, if p is zero, the initial impetus i will be conserved and, by the second principle of Aristotelian dynamics, the motion of the body will be rectilinear and uniform⁸⁸. Equation [3] works only if $P(t)$ is constant. However, this basic idea can be generalized for any continuous $P(t)$ by means of the concept of definite integral. If we take a small time interval dt_1 , we can regard $P(t)$ to be constant in that interval, so that we can apply [3] and we obtain:

$$[4] \quad I(t_1) = i + P(t_1)dt_1, \text{ where } t_1 \text{ is the instant at the end of the interval } dt_1.$$

The same obviously holds for a second interval dt_2 , for a third interval dt_3 , etc. The impetus at time t_n is thus given by:

$$[5] \quad I(t_n) = i + \sum_j P(t_j)dt_j, \text{ where } j = 1 \dots n$$

⁸⁸ If the resistance to move is assumed to be a constant.

Now, the limit of this sum for smaller and smaller intervals is the definite integral of $P(t)$ between time 0 and t . This justifies the following general hypothesis:

$$[I_2] \quad \mathbf{I}(t) = \mathbf{i} + \int_0^t \mathbf{P}(t)dt, \quad \text{where } \mathbf{i} \text{ is the initial impetus}$$

Equation-schema $[I_2]$ gives us the impetus at an arbitrary time as a function of the total propulsive force acting on the body. I will call the term $\int_0^t \mathbf{P}(t)dt$ *acquired impetus*⁸⁹. Schema $[I_2]$ will be called the *second principle of Impetus dynamics*, to distinguish it from $[I_1]$ which is *the first principle*. From $[I_2]$, $[2]$, $[I_1]$, and $[A_2]$ we finally obtain:

$$[6] \quad \mathbf{V}(t) = (\mathbf{vr} - \mathbf{p} + \int_0^t \mathbf{P}(t)dt + \mathbf{P}(t)) / \mathbf{R}(t), \quad \text{where } \mathbf{v}, \mathbf{r}, \text{ and } \mathbf{p} \text{ are, respectively, the initial velocity, resistance to move, and propulsive force}$$

This schema expresses the velocity as a function of the propulsive force and of the resistance of the body to move. $[A_1]$, $[A_2]$, $[I_1]$, and $[I_2]$ are the theoretical principles of Impetus theory. Even though $[6]$ is *the fundamental schema* of this theory, I do not take this to be a basic principle because it is derivable from the others.

The specific assumptions of Impetus theory depend on the particular phenomenon which is considered, and they are either specific principles or laws. Since $[A_2]$ relates the velocity to the total force and to the resistance of a body to

⁸⁹ Notice the formal identity between $\int_0^t \mathbf{P}(t)dt$ and our concept of impulse. However, this is not really impulse, because $\mathbf{P}(t)$ is not quite the Newtonian total force $\mathbf{F}_N(t)$. The reason why we cannot seriously identify $\mathbf{P}(t)$ with $\mathbf{F}_N(t)$ is that Impetus theory lacks the concepts of acceleration and inertial mass, and it also lacks the second principle of dynamics which connects these concepts to $\mathbf{F}_N(t)$.

move, we first of all need to determine which forces act on a body, and the time evolution of its resistance to move. By [I₁], to find the total force, we must determine the propulsive force and the impetus. Notice, however, that we need only specify the propulsive force. When this is known, the impetus is also known by [I₂]. Therefore, the general problem of Impetus theory can be put in the form: *specify the propulsive force which acts on a body, and its resistance to move.* How this problem is solved depends on the specific case.

For example, suppose that we want to explain the motion of a projectile. Then, the natural assumptions are that the propulsive force on the projectile in the horizontal direction is zero⁹⁰, and that the propulsive force in the vertical direction is equal to the weight. We thus obtain a specific principle:

$$[S_1] \quad P(t) = \langle 0 \ W(t) \rangle$$

We also assume that the resistance of the body to move is a positive constant:

$$[L_1] \quad R(t) = r, \text{ where } r > 0$$

and that the weight of the body is a negative constant:

$$[L_2] \quad W(t) = w, \text{ where } w < 0$$

Notice that the last two specific assumptions are laws, and that they are extremely simple. Also, the specific assumptions suggest themselves when a qualitative description of the phenomenon is considered. The three specific

⁹⁰ One might think that the propulsive force at time 0 is not null, because when the projectile is set in motion, there is a force acting on it. According to Impetus theory this initial force is not a propulsive force but, rather, is the initial impetus of the projectile which is equal to its initial velocity multiplied by its resistance to move. In fact, if we throw a stone, we are not *applying* a force to the stone at the moment of its release.

assumptions, together with the theoretical principles, imply that the motion in the horizontal direction has constant velocity and that the motion in the vertical direction is uniformly accelerated. In fact, from [S₁], [L₁], [L₂], and [6]:

$$[\lambda] \quad V(t) = \langle ((v_x r - 0 + 0t + 0) / r), ((v_y r - w + (\int_0^t w dt) + w) / r) \rangle = \langle v_x, (v_y + tw/r) \rangle$$

Impetus theory thus determines a theoretical framework for the phenomenon of projectile motion, and the model of this framework is the same as the model of the Newtonian framework obtained by assuming that weight and mass are constant. This particular case suggests that, with convenient translation rules, we may always transform a theoretical framework of Impetus theory into a Newtonian one, and conversely. Under certain restrictions, this is in fact true but, before proving this result, we need some further preliminaries.

5.2 The standard models of Impetus theory

I take a *standard model of Impetus theory* to be any dynamical model whose components are the positions and velocities of n bodies, where $n > 0$. Therefore, P is a standard model of Impetus theory, just in case $P = \langle \langle Y_1 V_1 \dots Y_n V_n T \rangle \langle Y_1(t) V_1(t) \dots V_n(t) V_n(t) \rangle \{Y_1 V_1 \dots Y_n V_n T\} \rangle$, where Y_i and V_i are, respectively, the position and velocity of body i . Notice that, given this definition of a standard model, *Impetus theory is not complete*. In fact, the set of magnitudes $T_i(P)$ determined by the theoretical principles T_i of Impetus theory is expressed by the following schemas: Y_p , V_p , P_p , F_p , R_p , and I_p where P_p , F_p , R_p , and I_p are, respectively, the propulsive force, the total force, the resistance to move,

and the impetus of body i . If $Y_i(t) = \alpha_i$ and $V_i(t) = \beta_i$ are given, the theoretical principles of Impetus theory do not allow us to deduce a law for P_i , F_i , R_i , or I_i . Hence, Impetus theory is not complete.

5.3 The theoretical principles of Newtonian dynamics

I take the theoretical principles of *Newtonian dynamics* to be the definition of velocity, the second principle of dynamics, and the definition of acceleration. The *definition of velocity* states that the velocity is the derivative of the position:

$$[N_1] \quad \mathbf{V}(t) = D_t(\mathbf{Y}(t))$$

The *second principle of dynamics* states that the acceleration of a body is directly proportional to the total force acting on it, and is inversely proportional to its mass:

$$[N_2] \quad \mathbf{A}(t) = \mathbf{F}(t) / \mathbf{M}(t), \text{ where } \mathbf{M}(t) > 0, \mathbf{A}(t) \text{ is the acceleration, } \mathbf{M}(t) \text{ the mass, and } \mathbf{F}(t) \text{ the total force acting on an arbitrary body.}$$

Finally, the *definition of acceleration* states that the acceleration is the derivative of the velocity:

$$[N_3] \quad \mathbf{A}(t) = D_t(\mathbf{V}(t))$$

The specific assumptions of Newtonian dynamics, as it happens for any other theory, depend on the particular system which is considered, and so they cannot be explicitly listed. It is for this reason that, to understand a theory, we need to look at those possible explanations which have become classic. Kuhn called these

typical frameworks "*paradigms*"⁹¹. Newtonian dynamics has many paradigms. One of the simplest is the framework for a harmonic oscillator. A harmonic oscillator consists of a body attached to a spring, where the only force acting on the body is the restoring force of the spring. This can be expressed as a specific principle:

$$[S_1] \quad F(t) = F_{bs}(t), \quad \text{where } F(t) \text{ is the total force on the body } b, \text{ and } F_{bs} \text{ is the force exerted on } b \text{ by the spring } s$$

The restoring force is proportional to the displacement of the body, so that we obtain a second specific principle:

$$[S_2] \quad F_{bs}(t) = -k Y(t), \quad \text{where } k \text{ is a constant, and } Y(t) \text{ is the position of } b, \text{ measured from its equilibrium point}$$

We also assume that the mass of the body is a constant. This can be expressed as a law:

$$[L_1] \quad M(t) = m$$

From [L₁], [S₂], [S₁], and [N₂] we obtain:

$$[E] \quad A(t) = -k/m Y(t)$$

Since the acceleration is the second derivative of the position, [E] is a differential equation of the second order, whose solutions, together with their first derivatives, determine the model $P = \langle\langle Y \ V \ T \rangle \langle Y[y \ v](t) \ V[y \ v](t) \{Y \ V \ T\} \rangle\rangle$. The pair $\langle A \ P \rangle$, where $A = \{N_1 \ N_2 \ N_3 \ S_1 \ S_2 \ L_1\}$ is thus a theoretical framework of Newtonian dynamics (the *harmonic oscillator framework*).

⁹¹ Or "exemplars". This is one of the two basic meanings which Kuhn gave to the term "paradigm". The other basic meaning refers to a whole *theory* which has become the received view for a scientific community, and is routinely used and developed by the members of this community (Kuhn 1970).

5.4 The standard models of Newtonian dynamics

I take the standard models of Newtonian dynamics to be the same as the standard models of Impetus theory, that is, the set Σ of *all dynamical models whose components are the positions and velocities of n bodies, where $n > 0$* . Notice that, given this choice of Σ , *Newtonian dynamics, like Impetus theory, is not complete*.

5.5 Two inconsistent theories which explain the same phenomena

It is surprising to realize that, *in a sense which will soon be clear*, Impetus theory and Newtonian dynamics are translatable, even though they contradict each other. That Newtonian dynamics and Impetus theory are inconsistent follows from their different interpretations of the concept of *total force*. The second principle of dynamics [N₂] affirms that the *total force* acting on a body is proportional to its *acceleration*. By contrast, the second principle of Aristotelian dynamics [A₂] states the proportionality between *total force* and *velocity*. The conjunction of [N₂] and [A₂] is inconsistent, as it can be quickly verified by considering the case of a body which moves with a constant velocity $v \neq 0$. From [N₂], we obtain $F(t) = 0$, but from [A₂] we obtain $F(t) = vR(t)$, where $R(t) \neq 0$. Therefore, $F(t) \neq 0$ and $F(t) = 0$.

It remains to be seen how we can translate Newtonian dynamics into Impetus theory and conversely. This is possible in the following sense. Let $\langle A_N P \rangle$ be a framework of Newtonian dynamics such that P is a standard model. A_N thus entails the position and velocity laws of all the bodies constitutive of P . Also

assume that the set of hypotheses A_N is equal to $T_N \cup \{ "M_i(t) = \alpha_i" \} \cup \{ "F_i(t) = \beta_i" \}$, where T_N are the theoretical principles of Newtonian dynamics, " $M_i(t) = \alpha_i$ " is a law for the mass of an arbitrary body constitutive of P , and " $F_i(t) = \beta_i$ " is a law for the total force of an arbitrary body constitutive of P . Under these hypotheses, it is possible to find an Impetus framework $\langle A_I, P \rangle$ whose set of hypotheses A_I is equal to $T_I \cup \{ "R_i(t) = \gamma_i" \} \cup \{ "P_i(t) = \delta_i" \}$, where T_I are the theoretical principles of Impetus theory, " $R_i(t) = \gamma_i$ " is a law for the resistance to move of an arbitrary body constitutive of P , and " $P_i(t) = \delta_i$ " is a law for the propulsive force of an arbitrary body constitutive of P . Furthermore, A_I entails the *same* position and velocity laws as A_N , so that $\langle A_I, P \rangle$ and $\langle A_N, P \rangle$ share the *same* model P . However, $A_I \cup A_N$ is inconsistent.

The converse is also true, in the sense that, if we start from the Impetus framework $\langle A_I, P \rangle$, then we can find the corresponding Newtonian framework $\langle A_N, P \rangle$, where $A_N \cup A_I$ is inconsistent. *The trick consists in using the appropriate translation rule.* In order to transform an Impetus framework into a Newtonian one (and conversely) we must *not* use the *natural* translation rule *total force* of Impetus theory = *total force* of Newtonian dynamics. Instead, we must identify the *propulsive force* of Impetus theory with the *total force* of Newtonian dynamics.

Let us first see how we can obtain a Newtonian framework $\langle A_N, P \rangle$ if we start from an Impetus framework $\langle A_I, P \rangle$. Since $P_i(t) = \delta_i$, from the second principle of Impetus dynamics we obtain: $I_i(t) = i_i + \int_0^t \delta_i dt$. From the first principle of Impetus dynamics, $F_i(t) = \delta_i + i_i + \int_0^t \delta_i dt$. Since $R_i(t) = \gamma_i$, from the second principle of

Aristotelian dynamics, $V_i(t) = (\delta_i + i_i + \int_0^t \delta_i dt) / \gamma_i$. From this, and the definition of acceleration, $A_i(t) = D_t(\delta_i + i_i + \int_0^t \delta_i dt) / \gamma_i$. Since we identify propulsive force and *Newtonian* total force, $F_i(t) = \delta_i$. Finally, from the second principle of dynamics $M_i(t) = \delta_i / D_t(\delta_i + i_i + \int_0^t \delta_i dt) / \gamma_i$.

Conversely, let us start from a Newtonian framework $\langle A_N P \rangle$. Since $F_i(t) = \beta_i$ and $M_i(t) = \alpha_i$, from the second principle of dynamics, $A_i(t) = \beta_i/\alpha_i$. From the definition of acceleration, $V_i(t) = v_i + \int_0^t \beta_i/\alpha_i dt$. Since we identify the *Newtonian* total force with the propulsive force, $P_i(t) = \beta_i$. Therefore, from the second principle of Impetus dynamics, $I_i(t) = i_i + \int_0^t \beta_i dt$, and from the first principle of Impetus dynamics, $F_i(t) = \beta_i + i_i + \int_0^t \beta_i dt$. Finally, from the second principle of Aristotelian dynamics, $R_i(t) = (\beta_i + i_i + \int_0^t \beta_i dt) / (v_i + \int_0^t \beta_i/\alpha_i dt)$. We have thus proved the following theorem:

Theorem 4 (Newtonian dynamics and Impetus theory are translatable and inconsistent)

- (a) Let $\langle A_N P \rangle$ be a framework of Newtonian dynamics such that: (i) P is a standard model; (ii) A_N is equal to $T_N \cup \{ "M_i(t) = \alpha_i" \} \cup \{ "F_i(t) = \beta_i" \}$, where T_N are the theoretical principles of Newtonian dynamics, " $M_i(t) = \alpha_i$ " is a law for the mass of an arbitrary body constitutive of P , and " $F_i(t) = \beta_i$ " is a law for the total force of an arbitrary body constitutive of P . Then, there is an Impetus framework $\langle A_I P \rangle$ such that A_I is equal to $T_I \cup \{ "R_i(t) = \gamma_i" \} \cup \{ "P_i(t) = \delta_i" \}$, where T_I are the theoretical principles of Impetus theory, " $R_i(t) = \gamma_i$ " is a law for the resistance to move of an arbitrary body constitutive of P , and " $P_i(t) = \delta_i$ " is a law for the propulsive force of an arbitrary body constitutive of P . Furthermore, $A_I \cup A_N$ is inconsistent (under the natural interpretation *total force* of Impetus theory = *total force* of Newtonian dynamics).
- (b) Let $\langle A_I P \rangle$ be a framework of Impetus theory such that: (i) P is a standard model; (ii) A_I is equal to $T_I \cup \{ "R_i(t) = \gamma_i" \} \cup \{ "P_i(t) = \delta_i" \}$, where T_I are the theoretical principles of Impetus theory, " $R_i(t) = \gamma_i$ " is a law for the resistance to move of an arbitrary body constitutive of P , and " $P_i(t) = \delta_i$ " is a law for the propulsive force

of an arbitrary body constitutive of P. Then, there is a Newtonian framework $\langle A_N, P \rangle$ such that A_N is equal to $T_N \cup \{ "M_i(t) = \alpha_i" \} \cup \{ "F_i(t) = \beta_i" \}$, where T_N are the theoretical principles of Newtonian dynamics, " $M_i(t) = \alpha_i$ " is a law for the mass of an arbitrary body constitutive of P, and " $F_i(t) = \beta_i$ " is a law for the total force of an arbitrary body constitutive of P. Furthermore, $A_N \cup A_I$ is inconsistent (under the natural interpretation *total force* of Newtonian dynamics = *total force* of Impetus theory⁹²).

proof:

see above.

I am now going to consider a consequence of the proof of this theorem which I take to be quite interesting. Let us first look at the relations between the concepts of Impetus theory and Newtonian dynamics. The concepts of *Aristotelian dynamics* are *position*, *velocity*, *total force*, and *resistance to move*. *Impetus dynamics* then adds the concepts of *propulsive force* and *impetus*. On the other hand, the concepts of *Newtonian dynamics* are *position*, *velocity*, *acceleration*, *total force*, and *mass*. Position and velocity are the same concepts as those of Aristotelian dynamics. Acceleration and mass are new concepts, while we have seen that the Newtonian concept of *total force* is in fact the concept of *propulsive force* in disguise. We can thus summarize the conceptual relations between Impetus theory and Newtonian dynamics as follows. Newtonian dynamics

⁹² However, if we introduce two *different* concepts of total force F_I , and F_N , which respectively are the total force of Impetus theory and the total force of Newtonian dynamics, then $A_I \cup A_N$ is consistent under the identification *propulsive force* = F_N . Even though this interpretation is logically possible, it is quite unnatural, for the idea that there are two different *total* forces acting on the same body sounds like a joke (this seems even more paradoxical when we realize that, since F_N = *propulsive force*, F_N is a *component* of F_I and, at the same time, F_N is a *total* force). Notice, however, that it is exactly this '*paradoxical*' interpretation which makes possible to translate an Impetus framework into a Newtonian one and conversely. It is also interesting to notice that looking at things from this point of view allows us to understand how the *conceptual shift* from Impetus theory to Newtonian dynamics is possible. More on this point later.

(i) *preserves* the old concepts of position and velocity; (ii) *introduces* the two new concepts of acceleration and mass; (iii) *reinterprets* the old concept of propulsive force by means of the two new concepts of acceleration and mass, and finally (iv) *eliminates* the three old concepts of impetus, total force, and resistance to move.

The interesting point is that it is possible to think of this result as being obtained by first *extending* Impetus theory, and by then *dropping* certain parts of this theory. The result of the extension step is a new theory whose conceptual structure is unwieldy. Once this extension is obtained, however, *the extended theory is simplified* by cutting those parts of the previous theory which have become superfluous. The final result of this process is a new theory *inconsistent* with the old one.

Let me explain. The *extension* of Impetus theory consists of two successive steps. The first is the introduction of the new concept of *acceleration* together with its *definition* [N_3]. The second, is the introduction of the new concept of *mass* together with the *second principle of dynamics* [N_2] which relates the new concepts of acceleration and mass to the old concept of propulsive force. *The theory thus obtained, even though consistent, is clumsy and conceptually redundant.* The propulsive force is in fact related to the velocity in two different ways. On the one hand, the velocity depends on the propulsive force through the concepts of impetus, total force and resistance to move⁹³. On the other hand, the velocity

⁹³ By the first principle of Impetus theory and the second principle of Aristotelian dynamics.

depends on the propulsive force through the concepts of acceleration and mass⁹⁴. Furthermore, either dependence can be eliminated without essentially changing the content of the theory, as theorem 4 has shown. Therefore, *there are two possibilities*. Either we drop the new concepts of acceleration and mass⁹⁵ and we stick to Impetus theory, or we eliminate the old concepts of impetus, total force, and resistance to move⁹⁶, and we thus produce Newtonian dynamics. I will call the first alternative "*the conservative move*", and the second one "*the revolutionary move*". Finally, notice that the revolutionary move leaves us with *only one concept of force* (the old *propulsive force*). This concept can thus be naturally interpreted as a *new concept of total force inconsistent with the old concept of total force*. Therefore, *the result of the revolutionary move is a new theory (Newtonian dynamics) inconsistent with the theory from which we started (Impetus theory)*. I am now going to tell a *story* which plausibly describes how the revolutionary move might have come about. Quite obviously, the heroes of my story are called "Galileo" and "Newton".

Impetus theory had two different problems. In the first place, *the theory lacked a definite formulation*. There were many different versions of the same basic

⁹⁴ By the second principle of dynamics.

⁹⁵ Together with the principles in which these concepts occur, that is, the definition of acceleration and the second principle of dynamics.

⁹⁶ Together with the principles in which these concepts occur, that is, the second principle of Aristotelian dynamics, and the first and second principle of Impetus theory.

ideas, and none of these was completely satisfactory from the formal point of view. There was no agreement on the *details* of the basic principles which relate velocity, total force, resistance to move, impetus and propulsive force⁹⁷. In the second place, *the theory had not provided so far any quantitative explanation of any phenomenon.*

Galileo was aware of both problems, and he was especially concerned with the second. He complained that nobody, so far, had set out to find the *mathematical laws of real* phenomena. He then decided to work on this problem, and in fact he produced the first complete theoretical frameworks for several concrete dynamical processes: his explanations of free fall, of the motion on an inclined plane, and of the motion of a projectile. Furthermore, he also developed experimental methods which gave a very strong empirical support to his mathematical models. In the course of his work, *Galileo realized that he could dispense with the concepts of total force, resistance to move, impetus, and propulsive force. This was accomplished by explicitly introducing the concept of acceleration.* Once this concept is clearly understood, hypotheses about accelerations are sufficient for deducing the laws of free fall, of the motion on an inclined plane, and of the motion of a projectile. He thus chose to present his explanations in this form⁹⁸. This move gave him a great tactical advantage, because *Galileo could thus avoid the*

⁹⁷ This is not surprising, because an adequate formulation requires the concept of definite integral.

⁹⁸ Galileo discovered the law of free fall in 1604, but he published the correct derivation only in 1638, in *Two New Sciences* (Galilei 1914).

problem of precisely formulating the theoretical principles of Aristotelian and Impetus dynamics. On the other hand, this also was the weak point of his explanations. They did not provide any explicit story about the forces involved or, as Galileo put it, they did not explain the cause of acceleration.

Then came Newton. Galileo had mainly focused on the empirical inadequacies of Impetus theory. In doing this, he had set aside the formal problems of this theory, and he had stressed the importance of the concept of acceleration. He had thus implicitly proposed a new theory whose only concepts were position, velocity, and acceleration, and whose theoretical principles were the definitions of velocity and acceleration. *This theory, however, had one basic problem: it did not say anything about the causes of motion, for any consideration of the forces involved had been eliminated.* Newton found an elegant way of solving this problem of Galileo's theory. *He reconsidered the concept of force within the new Galilean context which only involved kinematic concepts,* and he thus realized that *the old propulsive force* could be thought as the cause of *acceleration*. He then considered the *simplest* mathematical formula which relates acceleration to force, namely, the direct proportionality $\mathbf{A}(t) = (K \mathbf{F}(t))$, and he interpreted the proportionality term K as a *new concept analogous* to the *old* Aristotelian concept of resistance to move. That is, K , was to be understood as the inverse of the *resistance to change velocity*. The result of these thoughts was thus the second

principle of dynamics⁹⁹ $\mathbf{A}(t) = \mathbf{F}(t) / \mathbf{M}(t)$. Newton finally added this principle to Galileo's theory, and he thus created Newtonian dynamics.

Newtonian dynamics only contains *one* concept of force, which corresponds to what Impetus theory had called "propulsive force". Within Newtonian dynamics, however, this is a *new* concept, for the external forces acting on a body *directly* produce a change in its velocity while, according to Impetus theory, they first produce a change in the impetus which then produces a change in velocity. Furthermore, this new concept of force is the *only* concept of force needed to explain motion. Newton thus interpreted this concept as the *total* force acting on a body, so that his theory became *inconsistent* with Impetus theory. In this way, the conceptual shift from Impetus theory to Newtonian dynamics was complete¹⁰⁰.

At that time, the new theory also appeared to be clearly superior *on both formal and empirical grounds*. In the first place, its theoretical principles were explicitly stated as *mathematical principles*. By contrast, the theoretical principles of Impetus theory were never formulated in a sufficiently clear manner and, in

⁹⁹ The second principle of dynamics is the simplest hypothesis which can be formulated if one is looking for a relation between propulsive force and acceleration. In fact, the principle states the direct proportionality between propulsive force and acceleration. So, this really is the first thing that comes to mind if the goal is a mathematical relation between propulsive force and acceleration. One might then wonder why Galileo did not discover the second principle of dynamics. He probably did not take this problem seriously because he thought that forces were not necessary to explain motion.

¹⁰⁰ Obviously, Newton accomplished much more than this. However, the explicit formulation of the second principle of dynamics is the crucial point of the story. Without this principle, any explanation of motion in terms of forces was bound to be cast in the traditional framework of Aristotelian and Impetus dynamics. A similar interpretation of Newton's contribution is convincingly defended by Richard Westfall (1971, ch. VIII, ch. I, ch. III).

particular, they were never put in the form of mathematical principles. In the second place, *Newtonian dynamics was able to produce accurate quantitative explanations of real phenomena*. Due to its lack of a mathematical formulation, Impetus theory could not match these explanations. *In the presence of a better competitor*, the formal and empirical weakness of Impetus theory finally condemned it. The theory was thus abandoned, and its problems were never solved.

5.6 The heuristic values of Impetus theory and Newtonian dynamics

In principle, however, the problems which condemned Impetus theory could have been solved. In the first place, we have seen that it is possible to give an adequate mathematical formulation of the principles of this theory. In the second place, the lack of explanatory power was not due to an intrinsic lack of content but, rather, to the historical contingency that this content had not been made explicit.

Let me explain. We have seen that Newtonian dynamics is *translatable* into Impetus theory (see theorem 4). This implies that the content of Impetus theory includes all the models P of Newtonian dynamics determined by any set of hypotheses A_N equal to $T_N \cup \{ "M_i(t) = \alpha_i" \} \cup \{ "F_i(t) = \beta_i" \}$, where T_N are the theoretical principles of Newtonian dynamics, " $M_i(t) = \alpha_i$ " is a law for the mass of an arbitrary body constitutive of P , and " $F_i(t) = \beta_i$ " is a law for the total force of an arbitrary body constitutive of P . Even though this set of models may not be the whole

content of Newtonian dynamics, it is nevertheless a relevant fraction¹⁰¹. Furthermore, Impetus theory might even have an 'excess content', in the sense that some of its models might not be members of the content of Newtonian dynamics. Therefore, if we had to choose between Newtonian dynamics and *this version* of Impetus theory, we'd better consider other factors besides the content and the formal adequacy of the two theories. In fact, there is one basic reason why Newtonian dynamics should be preferred to Impetus theory: the *heuristic value* of Newtonian dynamics is much higher.

We have seen at the end of section 4 that the heuristic value of a dynamical theory $\langle \Pi \Sigma \{ \alpha \} T \Phi \rangle$ depends on the power of three sets of rules which are typically associated with specific components of the theory: **(i)** the rules associated with the definition of the set of standard models Σ - these rules allow us to select appropriate sets of *component magnitudes*; **(ii)** the rules associated with the theoretical principles T and their mathematical formulation - these rules allow us to select appropriate sets of *hypotheses*; **(iii)** the rules associated with the logic and mathematics of the theory $\{ \alpha \}$ - these rules allow us to *deduce a law for each component magnitude*. Let us then describe these three sets of rules for both Impetus theory and Newtonian dynamics, and compare their respective powers.

Since the standard models of the two theories are the same, the sets of *rules*

¹⁰¹ Notice that, by theorems 3 and 4, this set of models includes any model P such that **(1)** the components of P are the positions and velocities of n bodies; **(2)** P is specified by a framework $\langle A P \rangle$ of *classical mechanics* such that, for any body i constitutive of the model P , the specific hypothesis " $M_i(t) = m_i$ " is a member of A (where m_i is a constant and $M_i(t)$ is the mass of body i). Now, all the usual models of classical mechanics are in this set, for the mass of an arbitrary body is typically taken to be a constant.

for selecting component magnitudes of a given process or system ψ are also identical. These rules are: **(a)** determine n bodies whose positions and velocities might be components of a standard model of ψ ; **(b)** choose components $\langle Y_1, V_1 \dots Y_n, V_n, T \rangle$, where Y_i and V_i are, respectively, the position and velocity of body i . I take these two rules to be quite powerful. However, since they are the same for both theories, their power does not discriminate between the heuristic values of the two theories.

The same is true of the *rules for deducing a law for each component magnitude*. These rules depend on the logic and mathematics of the two theories. However, I have implicitly assumed that these two components are the same, namely classical logic and standard analysis. The deductive rules can thus be identified with all the heuristic rules of these two disciplines. Again, this set of rules is very powerful but, on this count, Newtonian dynamics and Impetus theory are on a par.

We are thus left with the sets of *rules for selecting appropriate hypotheses*. I take the standard set of rules of Newtonian dynamics to be the following: **(a)** choose the second principle of dynamics, and the definitions of acceleration, and velocity; **(b)** for each body, write a law which states that its mass is a constant; **(c)** for each body which is part of the model, individuate all the forces acting on it, and write a specific principle which states that the total force on that body is equal to the sum of all the forces acting on it. To simplify this task, keep in mind all the action-reaction pairs; **(d)** for each component force, try to determine

a simple law first or, if this fails, *a specific principle*. To simplify this task, keep in mind all the action-reaction pairs. *The right hand side of each specific principle should only contain the positions and/or velocities of bodies which are parts of the model.*

The heuristic rules listed above are expressly designed to obtain a system of differential equations of the second order. In fact, the second principle of dynamics states a relation between acceleration, mass, and force

$$[\text{N}_2] \quad \mathbf{A}(t) = \mathbf{F}(t) / \mathbf{M}(t),$$

and the other two principles of Newtonian dynamics imply that the acceleration is the second derivative of the position. Therefore, if n ($0 < n$) bodies are given, and *the total force and the mass of an arbitrary body i ($0 < i < n$) only depend on the positions and velocities of these bodies*, the second principle of dynamics determines a system of differential equations of the second order. The mass rule **(b)** implies that *any mass $M_i(t)$ is equal to a constant m_i* . The conjunction of rules **(c)** and **(d)** implies that the total force $F_i(t)$ of an arbitrary body i depends (at most) on the positions and velocities of the (other) bodies. It thus follows that the result of the application of these rules will always be a system of differential equations of the second order. Furthermore, the solutions of this system, together with their first derivatives, determine a dynamical model P.

The power of these rules is quite high for, first, *they often lead to correct results* and, second, *they are not too difficult to apply*. The power of these rules essentially depends on the fact that we are never *required to guess a law*. In fact,

the mass rule **(b)** explicitly tells us what the mass law of an arbitrary body is, namely, $M_i(t) = m_i$, where m_i is a constant. The force rule **(d)**, on the other hand, does suggest to look for a simple law of each component force. However, if this is not possible, this rule tells us to look for a *specific principle* which expresses the component force as a function of *positions and velocities*, and this is usually *easier* than guessing the correct form of the force law. Therefore, *the rules of Newtonian dynamics allow us to select hypotheses with a reasonable chance of being correct*. Impetus theory, on the other hand, does not fare as well on this count.

Recall first that the fundamental schema of Impetus theory [6] expresses the velocity as a function of the propulsive force and of the resistance to move. Therefore, in order to determine the velocity, we need some heuristic rule for determining the resistance to move. The basic problem, however, is this: *if the mass rule (b) of Newtonian dynamics is correct, then there cannot be any simple rule of Impetus theory which tells us a law for the resistance to move. In particular, the natural heuristic rule "take the resistance to move of an arbitrary body to be a constant" is almost always incorrect*¹⁰².

To see why, let us assume that the mass of an arbitrary body i is a constant, that is, $M_i(t) = m_i$. Let us then use schema [6] of Impetus theory to express $R_i(t)$ as a function of mass, velocity, and acceleration. We thus obtain:

$$[R] \quad R_i(t) = (m_i A_i(t) - m_i a_i + v_i r_i + \int_0^t m_i A_i(t) dt) / V_i(t)$$

¹⁰² However, this rule is correct if the acceleration is constant (see below). This, by the way, explains why Impetus theory works fine to generate frameworks for free fall, the motion of a body on an inclined plane, and projectile motion: in all these phenomena, the accelerations involved are constant.

We thus see that, if the mass is constant, the resistance to move depends on both the acceleration and the velocity. *Therefore, its law will in general have a quite complicated form.* Equation [R] thus implies that, in general, guessing a law for the resistance to move is at least as difficult as directly guessing a velocity law¹⁰³. We can thus safely conclude that *the heuristic value of Impetus theory is much lower than the heuristic value of Newtonian dynamics*¹⁰⁴, and that this lack of heuristic value is a decisive reason for not considering Impetus theory as a serious alternative to Newtonian dynamics.

¹⁰³ With one exception. If the acceleration is constant, then equation [R] implies that the resistance to move is also constant and equal to the mass.

¹⁰⁴ Unless there is some way of *not* guessing a *law* for the resistance to move. To avoid this step, one might try to guess a *specific principle* which expresses the resistance to move as a function of other magnitudes (for instance of the velocity). However, this will not in general work for, given the form of equation [6], we will not be able to write systems of differential equations.

Nevertheless, there is one specific principle which is always guaranteed to work, namely:

[S] $R_i(t) = (P_i(t) - p_i + v_i r_i + \int_0^t m_i D_t(V_i(t)) dt) / V_i(t)$, where m_i is a constant;

in fact, by substituting the right hand side of [S] for $R_i(t)$ in equation [6], and by differentiating both sides, we obtain $m_i D_t(V_i(t)) = P_i(t)$. But this is just the second principle of dynamics when the mass is taken to be a constant! Therefore, the existence of this method cannot count in favor of Impetus theory. It is rather a further reason why we'd better directly represent a dynamical problem in Newtonian dynamics.

It should also be kept in mind that all this discussion of the heuristic value of Impetus theory depends on the assumption that the mass rule of Newtonian dynamics is correct. If the mass were not constant, then taking the resistance to move to be constant *might* be a good strategy. To see in which cases this might work we can derive an equation similar to [R] for the mass. Let us then assume $R_i(t) = r_i$, and let us use equation [6] to express the velocity as a function of the propulsive force. By differentiating both sides of [6], we obtain $A_i(t) = (P_i(t) + D_t(P_i(t))) / r_i$. From the second principle of dynamics, and since we identify the propulsive force with the Newtonian total force, we finally obtain:

[M] $M_i(t) = r_i F_i(t) / (F_i(t) + D_t(F_i(t)))$

We thus see that, if the resistance to move were constant, then the mass would in general depend on the Newtonian total force and its first derivative. Also notice that equation [M] implies that $M_i(t)$ is constant if $F_i(t)$ is constant.

Chapter 4

COGNITIVE SYSTEMS AND THE MIND

1. Introduction
 - 1.1 The information processing theory of the mind
 - 1.2 Plan
 2. The problem of meaning
 - 2.1 The intension of mental states
 - 2.2 The extension of mental states
 - 2.3 The intentionality of mental states
 3. The relation between the body and the mind
 4. The problem of consciousness
 5. Is the mind a computational system?
 - 5.1 Lucas' argument
 - 5.2 Is the information processing theory of the mind consistent with the connectionist approach?
 6. Cognitive science as a branch of dynamics
 - 6.1 Cognitive systems
 - 6.2 Cognitive studies as dynamical studies
 - 6.3 The symbolic, the connectionist, and the dynamical approach
 7. Towards a dynamical theory of minds
-

1. Introduction

According to some authors, the goal of cognitive science is to understand human intelligence (Anderson 1990, 2). Other authors are more daring, they take the mind to be the object of their study (Johnson Laird 1988, part I). Even though these definitions are intuitively appealing, they need further specifications to be of some use. According to the first view, we should exclude from the domain of cognitive science any process which is not human or is not intelligent. The

problem with this prescription is its ambiguity. For example, should vision be an object of cognitive science? No, because vision is not typically human. Yes, because vision plays an important role in human intelligence. No, because vision is not an intelligent activity. Yes, because vision may be affected by some process in which intelligence is involved. This list of equally plausible arguments for excluding or including vision could be expanded as we like. If we just used a little imagination, we could easily produce similar lists for almost any other process or activity which we might consider. The situation with the second definition seems even worse, for the mind notoriously is an elusive object¹. Nevertheless, we should not be bothered too much by these skeptical reflections, for cognitive science, as any other science, simply *aims to explain the systems or processes in its domain*. At first sight, this definition may seem even less informative than the other two, but this impression is wrong. In the first place, this characterization gives us a sociological criterion for deciding whether a system should be considered cognitive. If this system has been studied by some member of the cognitive community, then we should certainly presume that it is a cognitive system. Second, and more important, this definition makes explicit that *the object of cognitive science is a specific set of natural systems or processes*. The hope is that we can learn more about some interesting general properties of the elements of this set. This is in fact the aim of this chapter.

¹ If this dismissal seems too quick to you, pay a little patience. I will discuss this view in due details when I turn to the analysis of the information processing theory of the mind.

1.1 The information processing theory of the mind

Newell and Simon maintain that the explanations of cognitive science are not in principle different from the explanations of any other science which is concerned with the dynamical behavior of some system². These and other authors assume that *the mind is a computational system realized by a concrete physical system*³, and that cognitive science studies the dynamical behavior of this computational system. Since any computational system is a dynamical system, it follows that the problem which a cognitive scientist is trying to solve is structurally analogous to the problems which are the usual focus of other sciences. For example, classical mechanics attempts to dynamically explain systems of a certain type, namely mechanical systems. We have seen in chapter 3 that a dynamical explanation of a system or process consists in specifying one of its dynamical models. The basic idea of the information processing theory of the mind is that cognitive science is doing something which is essentially similar: *it is trying to specify a dynamical model of a very complex system, namely the mind*. The way in which this problem is solved, however, depends on the particular nature of the system which cognitive science studies. Since, by assumption, the mind is a *computational* system and a very complex one, the natural way of studying this system consists in trying to write computer programs which emulate its behavior *in some specific type of*

² See Newell and Simon (1972, 11-12), and also Langley et al. (1987, 32-34).

³ I take a *physical system* to be any system which could be an object of physical study. Therefore, some, *but not all*, physical systems are *concrete*.

*situation*⁴. For example, we may be interested in studying how a subject learns a list of non-sense syllables. In the first place, we will gather all the relevant information about this task. This may involve reports of experimental studies, psychological, mathematical, or philosophical theories about the mind or some of its constitutive parts and, possibly, introspective reports of the mental activities involved in performing this task. On the basis of this information we will try to write a first sketch of a process which might lead to learn a list of non-sense syllables. This first sketch will usually take the form of a simple flow-chart⁵, so that the whole process is broken down in a number of simpler subprocesses. Then, we will try to add details to this first diagram. This is done by applying the same procedure recursively to each part of the diagram. For each subprocess, we will ask how its goal could be achieved. To come up with a possible solution for this problem, we will consider again all the relevant information, perhaps gathering new data if the ones we have are not sufficient for suggesting a plausible hypothesis. At the end of this work, we will have produced a more detailed sketch. We will then repeat the whole procedure until the diagram is so detailed that we can

⁴ Newell and Simon explicitly assume that the mind is a *universal computer*, very similar to a concrete digital computer. This means that the operations of the mind will depend on the programs which are stored in its memory. Different programs specify different computational systems, and these systems have particular cognitive abilities appropriate for specific situations or tasks. Therefore, *the aim of a concrete empirical research is to specify a dynamical model of one of these special purpose computational systems*. This can be obtained by writing a computer program which specifies a second computational system, and this computational system must *emulate* the special purpose computational system which we want to explain.

⁵ Any other equivalent representation, obviously, will do. The important point is that the whole process be broken down in simpler parts, and that the order in which these subprocesses are executed be univocally specified.

translate it in some computer language. Once we have a running program, we can claim that we have a possible explanation of how a list of non-sense syllables may be learned. At that point, we will test the performance of our program against the performance of some subject in the same task. If the match is sufficiently close, we may provisionally conclude that we have explained the dynamics of the subject's mind in that particular task⁶.

This view of the aims and methods of cognitive science seems to be on the right track. Nevertheless, it leaves open some serious problems. This theory stands on the basic assumption that *the mind is a computational system realized by a concrete physical system*. This methodological principle⁷ raises at least five different types of question. First, how do we explain the *intentional aspects of our mental life*? This question can also be put in the more concise form: how do we explain *meaning*? Second, how do we explain *the relation between the mind and the underlying physical (chemical, biological) structure* which realizes it? This is a contemporary version of what philosophers traditionally call "the mind-body problem". Third, how do we explain the *subjective aspects of mentality*? Or, equivalently, how do we explain *consciousness*? Fourth, why assume that the

⁶ A clear statement of the methods involved in constructing computational explanations of mental processes can be found in Feigenbaum and Feldman (1963, 269-76).

⁷ Some cognitive scientists maintain that this principle has been *discovered*, and that it is a *theoretical* principle like, for instance, the law of universal gravitation. Be that as it may, there is little doubt that this principle is also taken to be the foundation of the correct methods of cognitive science. The basic question I am interested in is whether this principle can justifiably function as a methodological one.

mind is a *computational system*? Could the mind be a system which is not computational? This question can also be rephrased: could the mind be a system which cannot be emulated by a Turing machine? Could the mind be essentially more powerful than, or essentially different from, a Turing machine? Fifth, why assume that *the mind* is a computational system? This question may seem identical to the fourth one, but notice the different emphasis. The fourth question does not challenge the fact that we need some assumption about the mind to justify the aims and methods of cognitive science. It only challenges the specific form of this assumption. The fifth question, instead, is much more radical. It may be rephrased: *could the primary object of cognitive science be something different from the mind, and could this something not necessarily be computational?*

These five questions are serious ones. I will discuss each of them in turn. My analysis will show that the information processing theory of the mind can provide very good answers to the first three questions, but that it is not able to solve the fourth problem. I will then consider the implications of the fifth question, and I will finally propose an alternative methodological view - *the dynamical approach* - which I take to be superior to the information processing theory of the mind.

1.2 Plan

This chapter is divided in six further sections. The second section deals with the problem of meaning. I first consider an abstract version of Searle's Chinese

room argument (1980, 1984, 1990a), which supposedly shows that the information processing theory of the mind cannot ascribe intentionality to mental states. Searle maintains that this conclusion follows from two basic facts: (1) syntax is not sufficient for semantics; (2) the information processing theory of the mind identifies mental states with *completely uninterpreted* strings of symbols. My reply challenges the second premise. While it is true that mental states are identified with strings of symbols which may not have a *reference*, it is false that these strings lack *any* meaning. In fact, any such string can always be thought as a *sentence* of a *theory*, and any such sentence has an *intensional meaning* which is determined by the theorems of the theory⁸. I then argue that the information processing theory of the mind can also *in principle* deal with the problem of the *referential content* of mental states.

The third section discusses the nature of the relation between the body and the mind⁹. Searle (1990b) has claimed that a physical system cannot be said to realize a computational one in an objective sense. If Searle were right, the information processing theory of the mind would face a serious problem, for the proponents of this approach maintain that the mind is a computational system realized by the brain, and they take this hypothesis to be a *synthetic* one. I have

⁸ This, obviously, is not a new idea. However, my argument shows how this view has a simple and, I believe, natural interpretation in terms of the accessibility relations between the possible states of a computational system.

⁹ What I intend is: *if we identify the mind with a computational system*, can we describe the relation between the body and the mind in an objective and precise manner? Searle (1990b) maintains that this is impossible.

shown in section 6 of chapter 1 that the realization relation between two dynamical systems does not involve any conventional element, for it depends on the *existence* of a mapping between the *states* of the realized system and *sets of states* of the realizing one. The relation between the mind and the brain postulated by the information processing theory is in fact a special case of the realization relation between two dynamical systems. I thus conclude that Searle's claim about the conventional character of this relation is unjustified.

The fourth section is dedicated to the problem of consciousness. The thesis I defend is that the information processing theory of the mind can *in principle* explain the *nature* of conscious states, and I also suggest that a detailed hypothesis constructed along the lines I sketch might be empirically tested.

The fifth section discusses whether the identification of the mind with a computational system is justified. I first consider Lucas' classic argument (1961, 1968a, 1968b) which has been recently revived in a slightly different form by Penrose (1989, 108-12, 416-18). According to Lucas, we could deduce a contradiction from Gödel's incompleteness theorem for arithmetic conjoined with two other premises: (i) the mind is a computational system; (ii) if the mind is a computational system, then it can be identified with a theory which satisfies the conditions of Gödel's Theorem. Lucas then interprets his argument as a *reductio* of the first premise, and he thus concludes that the mind is not a computational system. My reply points out that the second premise of Lucas' argument is false if the first premise is true. In fact, all the theories to which Gödel's theorem

applies are *not decidable*, but any computational system can be identified with a *decidable* theory. My point is based on the definition of a computational system which I have given in chapter 1.

I then discuss whether the identification of the mind with a computational system is consistent with the connectionist approach to cognitive science. I argue that it may not be, provided that the mind can be identified with a finite neural net whose units have continuous activation levels. Furthermore, the question of whether this identification is possible is an *empirical* one.

The conclusion of the fifth section implies that the hypothesis of the computational character of the mind is a bold step, possibly a 'misstep'. In the sixth section, I propose a new view of the aims of and methods of cognitive science which does not presuppose this assumption. The basic idea of this methodological theory is that cognitive science attempts to explain a particular set of *dynamical systems*, which I call "*cognitive systems*". Therefore, cognitive science can be thought as a special branch of *dynamics*. The *dynamical approach* which I propose is characterized by three basic tenets: (a) the explicit recognition of the dynamical character of cognition; (b) the disposition to apply methods and concepts from dynamical system theory to the study of 'the mental'; (c) the disposition to explore the *whole space* of the dynamical systems in order to locate and map the region of the *cognitive systems*.

Finally, the seventh section discusses some implications of the dynamical approach for a theory of minds. I state four general properties of minds, and I

then ask whether a *dynamical theory* can *in principle* explain these properties¹⁰. My thesis is that it can, and that these explanations are at least as good as those provided by the information processing theory of the mind. In fact they are superior, for they do not *presuppose* the *computational* character of the mind, but only the *weaker hypothesis* that minds are *dynamical* systems realized by concrete physical systems.

¹⁰ These properties are: minds **(1)** are realized by concrete physical systems; **(2)** have intentional states; **(3)** have conscious states; **(4)** are cognitively universal. By a *dynamical theory of minds* I intend any theory which accepts the hypothesis: minds are *dynamical* systems realized by concrete physical systems.

2. The problem of meaning

In a now classic series of articles, Searle put forth an elegant argument which clearly shows the deep significance of the first question¹¹ (Searle 1980; Searle 1984, ch. 2; Searle 1990a). The goal of this argument is to prove that it is impossible to ascribe intentionality to mental states if we identify them with computational states. The argument, in its simpler form, goes like this. Mental states are intentional, therefore meaning is essentially involved in them. Computational states, on the other hand, are purely syntactical entities, because they can always be identified with strings of uninterpreted symbols. Therefore, if we identify a mental state with a computational one, it does not have any meaning. It thus follows that mental states are not intentional. This contradicts the hypotheses. We must thus conclude that a mental state is not a computational state¹².

This is a clever argument and, even though many replies have been devised, none of them is completely adequate. Notice first that the argument, in this simple form, is certainly valid. This means that some replies will not work against this form. One of these is the so called "system reply". This reply shows that the

¹¹ Recall that the first question asks how we can explain intentionality (or meaning) if we identify the mind with a computational system.

¹² Notice that this conclusion implies that the mind is not a computational system, for the identification of a mental state with a computational state follows from the identification of the mind with a computational system.

original form of Searle's argument¹³ involves a *non sequitur* because the assumptions of the original argument only warrant the conclusion that *one part*¹⁴ of a computational system cannot have intentionality, while Searle should deduce this conclusion for the *whole system*. Clearly, this type of counter attack will not work against the abstract form of the argument. In this form, there is no confusion between a system and its parts. Intentionality is ascribed to mental states in the premises, and the conclusion denies this attribution. This is a real contradiction, and there is no formal way of dismissing it. The only possible replies are those which question whether the premises are true, or at least probable.

We certainly do not want to deny the truth of the first premise. That mental states are intentional is one of those facts which are rooted in our most basic intuitions. From the logical point of view, we could still reject these intuitions, but it is sensible to require that this move should be our last resort. We are thus left

¹³ In the original form of the argument, Searle (1980) imagines that a "Chinese understanding program" has been written. A computer equipped with this program 'understands' Chinese in the sense that it is able to carry on a Chinese conversation, and that its Chinese output cannot be distinguished from the output of a native Chinese speaker engaged in a similar conversation. That is, this computer 'understands' Chinese in the sense that it is able to pass a Turing test (Turing 1956) for understanding Chinese. Searle further assumes that the program consists of a series of rules which, to any possible Chinese input associate an appropriate Chinese output. Searle then imagines to simulate the operations of the *computer's interpreter*, which reads the Chinese input, looks for the appropriate program instruction, interprets it, and then produces an appropriate Chinese output. Searle then claims that going through all this process does not enhance his capacity of understanding Chinese, which is zero to start with, and zero after he has performed the purely syntactical operations prescribed by the Chinese understanding program. Searle thus concludes that the *computer* equipped with the Chinese understanding program does not understand Chinese, because the computer performs exactly the same operations which he has performed, and he has not acquired any Chinese understanding by performing these operations.

¹⁴ Namely, the computer's interpreter (see footnote 13) which manipulates input Chinese symbols according to a specified set of rules (the Chinese understanding program) and then outputs other, appropriate, Chinese symbols.

with the second premise. This premise affirms that computational states are strings of *uninterpreted* symbols. As many people have remarked, this is not true in general. This reply is known as "the robot reply". Suppose a computer is given sensors and effectors through which it can interact with the external world. Then, through the chain of causal interactions in which the computer is involved, some of its symbols may come to represent external objects. Since these symbols now represent external objects, they have a meaning or, in other words, they are *interpreted* symbols. The problem with this reply, though, is that it is too specific. What if the computer does not have sensors and effectors, or if it is not involved in causal interactions which confer an external reference to its symbols? Should we then agree with Searle and conclude that, in this case, computational states have no meaning? Many people in the cognitive science community are willing to accept this conclusion. But the problem with it is that it is at odds with the functional approach which they also accept.

2.1 The intension of mental states

To see why, we must go back to Frege's distinction between two aspects of meaning: the *extension* of a symbol, and its *intension*¹⁵. The extension is the entity to which a symbol refer. For instance, if we consider the description "the morning star", its extension is Venus. Obviously, two different symbols may have

¹⁵ I use the adjective "intentional" to refer to that aspect of mental states which involves meaning or semantic content. As usual, I then divide meaning into two components: extension and intension. The intentional character of mental states thus includes both extensional and intensional meaning.

the same extension. For example, both symbols "Venus" and "the morning star" refer to the planet Venus. However, these two symbols do not have the same meaning. In fact, we usually take "the morning star shines in the morning" to be true of the morning star but, if we do not know that the morning star is Venus, we may not take "Venus shines in the morning" to be true of Venus. Therefore, what we know about Venus may differ from what we know about the morning star. But this would be impossible if "Venus" and "the morning star" had exactly the same meaning. On the other hand, "Venus" and "the morning star" have the same extension. We must thus conclude that the complete meaning of at least one of these two symbols has an additional part. Let us call this additional meaning the intension of a symbol. The problem now is: where do intensions come from?

We can take at least two different approaches to answer this question. First, we can assume the existence of a subject which confers intensions to symbols. This is the metaphysical stance which is preferred by some philosophers. Second, we can simply regard *intensions as coming from the relations which a symbol bears to other symbols*. The basic idea is that, whenever a set of symbols has a theoretical structure defined on it, it is this structure which confers an intension to its elements. Another way of expressing the same idea is: it is the functional relation of a symbol to other symbols in a conceptual system which produces its intensional meaning. Going back to Frege's example, this means that, to understand where the intensions of "the morning star" and "Venus" come from, we should consider the conceptual system in which these symbols are embedded.

In this case, this system can simply be identified with the deductive closure of all known (astronomical) facts. This deductive closure, $C(K)$, can be expressed as a set of sentences and, among these sentences, there are some which contain the symbol "the morning star" and some which contain the symbol "Venus". Let M be the first set, and V be the second one. Intuitively, M and V are all the known facts which explicitly contain the symbols "the morning star" and "Venus". The complete intensional meaning of these two symbols can thus be identified with the deductive closures¹⁶ of M and V , which I indicate by " $C(M)$ " and " $C(V)$ ". Presumably, we know that the morning star shines in the morning, so that the sentence "the morning star shines in the morning" is in $C(M)$. However, if our conceptual system does not also contain "Venus = the morning star", the deductive closure of the set associated with "Venus" may differ from $C(M)$, and this explains how the intensional meanings of the two symbols may be different. This also explains why, if we do not know that the morning star is Venus, we may not take "Venus shines in the morning" to be true, for this sentence may not follow from any other known fact in K .

This sketchy view of intensional meanings makes also clear what we intend when we say that the axioms of a theory implicitly define its terms. The classic example is the partial definition of the primitive terms "point" and "line" which is provided by the axioms of Euclidean geometry. One of these axioms is: for any

¹⁶ The deductive closure of M (or V) will in general contain sentences in which "the morning star" (or "Venus") does not occur. However, these sentences are parts of the intension of "the morning star" (or "Venus") for they are 'implicitly contained' in our explicit knowledge about the morning star (or "Venus").

two points, there is exactly one line which passes through them. Now, let us simply take the intension of "point" ("line") to be the deductive closure of the set of all geometric theorems in which this term occurs. Since any axiom is a theorem, the intension of "point" ("line") contains the axiom "for any two points, there is exactly one line which passes through them". Therefore, the intensional meaning of either term is partially determined by this axiom. Definitions also contribute to the meaning of the terms which they contain, because all definitions can always be considered to be new axioms¹⁷. In general, *the intension of a symbol can always be identified with the deductive closure of the theorems (or definitions) in which that symbol occurs*¹⁸. Analogously, *the intension of a sentence is its deductive closure, that is, all other sentences which that sentence implicitly contains*.

The idea that the meaning of a symbol is (partially) determined by the theoretical system of which this symbol is a part is not new. For example, we have just seen that the interpretation of the axioms of a formal system as partial definitions of its primitive terms expresses the same idea. Carnap's view of

¹⁷ Notice an interesting consequence of this view. Suppose a predicate P is introduced by means of the definition "x is a P iff $\alpha(x)$ ". Then, this definition contributes to the meaning of P *and of all primitive or defined terms which occur in $\alpha(x)$* . This is why adding definitions to a theory is not a purely formal expedient. A new definition *expands the meaning of the terms which are already in the theory*. Obviously, a definition is not creative. This, however, only implies that the additional meaning of an old term must always involve the newly defined term. That is, what you learn when you define a new term is *a new way of expressing old facts*: facts stay the same, but their meaning changes.

¹⁸ We can take this to be true even for logical symbols. If a logical symbol is a part of a theory, then it has intensional meaning, even though it does not have any extension. Its intension is completely determined by the set of theorems in which it occurs.

analytic truth is also a version of this idea, for the meaning of a term is determined by a subset of the axioms of a theory (the meaning postulates). Feyerabend (1975) and Kuhn (1970) have advocated an extreme version of the same idea, according to which the meaning of a term is *completely* determined by the theory of which it is a component. Finally, the implications of this view for the philosophy of mind have been variously expounded by several authors (see, for example, Churchland 1979, ch. 3; Churchland 1988, 31, 63; Rapaport 1988; Tye 1989, 67-75).

Still, my version of this view may not seem to solve the problem of the intension of mental states. If we identify mental states with the states of a computational system, any mental state is a string of symbols in a specified set. Since this set is the phase space of a dynamical system, there is a structure defined on it. But this structure may not seem to be a theory, so that we may not conclude that mental states have intensions. This, however, is a false impression, for *any computational system can be identified with a theory*. Recall that a computational system can be identified with a discrete dynamical system (a cascade) $\langle Z M H \rangle$, where Z are the integers (or the non-negative integers), M is a decidable set of finite strings of symbols, and H is a function from M to M which can be computed by a Turing machine¹⁹. Among other things, this means that this system can be identified with a theory. The axioms of this theory are all the strings in M , and its inference rules are the transition function H and the identity

¹⁹ See the definition of a computational system (ch. 1, def. 3).

function on the phase space M . Therefore, if we identify the mind with a computational system $\langle Z M H \rangle$, any mental state $x \in M$ is a theorem, so that x has an intensional meaning²⁰: namely, the set of all other mental states which can be deduced from x . This set is equal to the orbit of x . We can thus conclude that, if the mind is a computational system, then *the intensional meaning of a mental state x is the orbit of x in the mind's phase space.*

We can now make explicit the implications of my view for Searle's argument. If we identify a mental state with a state of a computational system, *it is false that this string of symbols does not have any meaning*²¹. It may not have an extension, but it certainly has the intension determined by the structure of the phase space. In conclusion, the information processing theory of the mind can deal very well with one aspect of the problem of the intentionality of mental states, namely, with their *intensional* meaning. We must now consider the *extensional* aspect.

2.2 The extension of mental states

We have seen above that, if the mind is a computational system, then any

²⁰ Any theorem of a theory is a *sentence* of that theory, and I have proposed to identify the intensional meaning of a sentence with its deductive closure. Therefore, the intensional meaning of a theorem is the set of all sentences which can be deduced from that theorem.

²¹ Searle claims that his argument is a decisive refutation of the information processing theory of the mind because it is based on the logical fact that *syntax* is neither constitutive of, nor sufficient for, semantics (1990a, 27). We may concede that this is a logical fact, but it is irrelevant. What Searle has not considered is that a *theory* is sufficient for one aspect of semantics, namely for determining *intensional* meaning. Obviously, a theory is not sufficient for the other aspect of semantics, that is, *extensional* meaning. I will discuss this point below.

mental state can be identified with a finite string of symbols, and this string of symbols is a *sentence* of the theory whose axioms are all the possible states of the system, and whose inference rules are the transition function of the system and the identity function on the phase space. Any mental state thus has an *intensional meaning* which is the set of all other mental states which can be deduced from it. The question I am now going to address is whether this view also allows us to attribute an *extensional meaning* to mental states. By now, it should be clear that the identification of the mind with a computational system is *not sufficient* for determining the extension of mental states. Since any mental state can be thought to be a sentence, its extension must be a truth value: *true*, if the state of affairs expressed by that sentence occurs, *false* otherwise. The problem, however, is that *nothing ensures that this sentence does in fact express a state of affairs*, for the *reference* of its constitutive parts has not been fixed so far.

It might seem that a quite simple solution of this problem is available. The information processing theory of the mind identifies the mind with a computational system *realized by a concrete physical system*. This concrete physical system can causally interact with its environment, so that these interactions may determine an external reference for the constitutive parts of (some) mental states. Therefore, (some) mental states may, after all, come to express states of affairs. It thus follows that each of these mental states has an extension: *true*, if the state of affairs which it expresses occurs, *false* otherwise.

However, it is not difficult to realize that this solution of the problem of the extension of mental states does not work. First, we need to look at the general conditions which a string of symbols must satisfy in order to express a state of affairs. These conditions can be synthesized as follows: **(i)** the string of symbols has different *parts of specified forms* which are arranged in a *grammatically correct* way; **(ii)** the constitutive parts which *may* bear a referential relation to other objects have been specified. Let us call these parts of a string of symbols its *descriptive parts*; **(iii)** the reference of all the descriptive parts has been fixed.

Second, we must consider whether the hypothesis that the mind is a computational system realized by a concrete physical system ensures that mental states satisfy conditions **(i)** and **(ii)**. As I have remarked earlier, this hypothesis allows us to identify a mental state with a string of symbols which is a sentence of a certain theory. However, this string of symbols is a *sentence* only in the sense that it bears specified *inferential relations* to other sentences. Therefore, *this sentence may not have the kind of grammatical structure which enables it to express a state of affairs*. In fact, the *syntactic form* of this sentence must only satisfy the quite general property of being a string of symbols of a *decidable type*, and this property *does not* entail the full fledged compositional syntax required by conditions **(i)** and **(ii)**.

We thus see that the identification of the mind with a computational system realized by a concrete physical system does not imply that mental states have the kind of compositional structure required by conditions **(i)** and **(ii)**, that is, a

syntactic structure which enable them to express states of affairs when the reference of their descriptive parts is fixed. This is why the problem of the extension of mental states cannot be solved along the simple lines proposed above. This solution assumes that *the causal interactions between the environment and the concrete physical system which realizes the mind confer an external reference to the descriptive parts of (some) mental states*. It then concludes that, since the reference of each descriptive part of these mental states has been fixed, they now express states of affairs²². But this argument relies on a false premise, for it implicitly presupposes that the identification of the mind with a computational system realized by a concrete physical system entails that mental states have a full fledged compositional syntax²³.

Nevertheless, there is an obvious way for solving this problem: we can simply *assume that mental states have a full fledged compositional syntax*. According to this view, mental states are sentences of a special language²⁴ which satisfies conditions **(i)** and **(ii)**. It thus follows that mental states *may* express states of affairs. Furthermore, if the concrete physical system which realizes the mind is engaged in a series of causal interactions which ascribe a reference to the descriptive parts of (some) mental states, these mental states *will* express states of affairs. Therefore, each of these mental states has an extension: *true*, if the

²² Therefore, each of these mental states has an extension: *true*, if the state of affairs which it expresses occurs, *false* otherwise.

²³ That is, that they satisfy conditions **(i)** and **(ii)** above.

²⁴ In *The language of thought*, Fodor (1975) advocates a similar position.

state of affairs which it expresses occurs, *false* otherwise.

2.3 The intentionality of mental states

We have seen in the previous two sections that *the identification of the mind with a computational system realized by a concrete physical system* implies that *all mental states have intensional meaning*. However, this hypothesis is not sufficient for ascribing *extensional meaning* to mental states. In order to solve this problem, two other hypotheses have been proposed: first, that *mental states have a full fledged compositional syntax*; second, that *the concrete physical system which realizes the mind is engaged in a series of causal interactions which confer a reference to the descriptive parts of (some) mental states*. These two hypotheses, conjoined with "the mind is a computational system realized by a concrete physical system", entail that *(some) mental states have extensional meaning*.

The question I am now going to address is whether we can solve the problem of the intentionality of mental states in a simpler way. To start with, let me explicitly say what I want to achieve. *My goal is to outline an explanation of the intentionality of mental states which assumes the hypothesis that the mind is a computational system realized by a concrete physical system, but does not assume the hypothesis that mental states have a full fledged compositional syntax*. My proposal for solving this problem is as follows. In the first place, I deny that an adequate explanation of the intentionality of mental states must imply that

(some) mental states have extensions. That is, I deny that the explanandum "mental states are *intentional*" is equivalent to "all mental states have intensional meaning, and (some) mental states have extensional meaning". Rather, I maintain that "mental states are *intentional*" should be interpreted as the weaker assertion "all mental states have intensional meaning, and (some) mental states have *referential content*²⁵". In the second place, I claim that, under this interpretation, the assertion "mental states are intentional" follows from just two hypotheses: **[1]** the mind is a computational system realized by a concrete physical system; **[2]** the concrete physical system which realizes the mind is engaged in a series of causal interactions which confer a reference to some parts of (some) mental states.

Let me say first what I intend by "having *referential content*". *An object or entity has referential content just in case it has parts, and some of its parts bear a referential relation to other objects or entities.* This definition implies that everything which has an extension also has a referential content, however, the converse is false. There are things which have a referential content but do not have an extension. This is clear, for an object which does not have an extension has a referential content if one of its *proper parts* has an extension. As a concrete example, think of a sheet of paper on which your name is written. Your written name is in a referential relation to you and, obviously, it is a part of the sheet of paper on which it is written. Therefore, this sheet of paper has a referential

²⁵ See below for the definition of *referential content*.

content. However, it does not have an extension, for what has an extension is your written name, not the whole sheet on which your name is written.

I have proposed above that "mental states are intentional" should be interpreted as the conjunction of: **(a)** all mental states have intensional meaning; **(b)** (some) mental states have referential content. In addition, I have claimed that **(a)** and **(b)** follow from the hypotheses: **[1]** the mind is a computational system realized by a concrete physical system; **[2]** the concrete physical system which realizes the mind is engaged in a series of causal interactions which confer a reference to some parts of (some) mental states. I am now going to give an argument to support this claim. We have seen in section 2.1 that **(a)** follows from **[1]**. Hypothesis **[1]** also implies that a mental state can be identified with a finite string of symbols, and *any string of symbols has parts*, namely the set of all substrings of that string. Therefore, hypothesis **[2]** (together with the definition of referential content) implies that (some) mental states have referential content, that is, **(b)**.

Let me finally summarize the results of this section. Searle has claimed that the information processing theory of the mind cannot solve the problem of the intentionality of mental states. Contrary to Searle's claim, I have shown that the identification of the mind with a computational system realized by a concrete physical system implies that all mental states have intensional meaning. Furthermore, I have also shown that the information processing theory of the mind can *in principle* solve the problem of the *referential content* of mental states. This

solution is based on the further hypothesis that the concrete physical system which realizes the mind is engaged in a series of causal interactions which confer a reference to some parts of (some) mental states²⁶. Obviously, the *details* of this hypothesis *should* be worked out, but I do not see any reason why, *in principle*, this research program should be doomed to fail. In fact, some recent work along these lines seems to be quite promising²⁷.

²⁶ I have formulated this hypothesis in a very general form, so as not to beg any question as regards the *specific nature* of (i) the *mechanisms* which assign a reference to parts of mental states; (ii) the *types of objects* to which these parts refer. This hypothesis is thus compatible with a *sophisticated theory* of referential content such as, for instance, Harnad's symbol grounding proposal (1987, 1989, 1990). According to Harnad, the referential content of the symbolic level has a complex structure. Elementary symbols are names for object and event categories. The reference of elementary symbols is fixed (or grounded) by means of non-symbolic representations of two kinds: iconic representations and categorical representations. Furthermore, there is a second type of symbols (complex symbols) which express relations between object or event categories.

²⁷ See, for example, Harnad's theory of symbol grounding (1987, 1989, 1990). See also Korb (1991).

3. The relation between the body and the mind

I turn now to the second question: how can the information processing approach explain the relation between the mind and the underlying medium which supports it? This is an intriguing question and, as Searle has pointed out (1990b), it is surprising that those who favor the computational approach have not bothered giving this question a precise answer. Searle certainly is right in claiming that nobody, so far, has come up with an adequate solution of this problem. But he is wrong in thinking that there are some basic facts about the nature of a computation which preclude an objective solution of this problem. I will come back to this issue later. For the moment, let me just remark that the difficulty of this problem is not due to some arcane property of a computational system²⁸ but, rather, to the fact that the *realization* relation between two *arbitrary* dynamical systems has not been studied with the due care. I have given in chapter 1 an adequate analysis of this relation. I will now show that this abstract analysis is all that the information processing approach needs to explain the relation between the mind and the underlying medium²⁹. I will try to keep technical details to a minimum. For the formal analysis of the realization relation the reader should refer

²⁸ According to Searle, nothing is a computer if there is not a subject which interprets the workings of the system as computational operations (1990b, 28). This is just a misunderstanding. There is nothing which is subject-relative in the nature of a computer. Computers are dynamical systems with a special structure, just like physical systems are. If Searle is really serious about this view, then he should also maintain that physical systems are subject-relative. Why not say that nothing is a physical system if there is no subject which interprets the workings of the system as transitions from one physical state to another physical state?

²⁹ Of course, under the assumption that the mind is a computational system. Searle (1990b) concedes this premise and then attempts to show that there is no objective sense in which a physical system can be said to realize a computational one.

to chapter one.

In the first place, it is important to make clear that the relation between a computational system and the concrete system which implements or realizes it does not depend on the particular nature of these two systems but, rather, on the fact that both systems are deterministic dynamical systems. That a computational system is a deterministic dynamical system is true by definition. We could still question whether we are always justified in taking the hardware of a computer to be a deterministic dynamical system. For, if we go down to the level of quantum mechanics, this assumption may not be tenable³⁰. Nevertheless, given any medium size concrete system, this system obeys the laws of classical physics. Therefore, it is always possible to identify such a concrete object with a classical physical system. In particular, this is also true for the brain or, more in general, for the nervous system, the two types of hardware which we are concerned with. Now, according to the information processing approach, the mind is a computational system. It thus follows that the mind-body problem will be solved if we can explain the sense in which a dynamical system (brain, or nervous system) realizes, or implements, a second dynamical system (mind). Of course, it should be kept in mind that this is all that there is to the mind-body problem only if we have identified the mind with a computational system. Still, even in this form, the solution of this problem is not obvious.

³⁰ There are some interesting speculations on the nature of 'quantum computers' (Deutsch 1985). The relation between computation theory and quantum mechanics is a fascinating field, but I will be content with explaining the relation between systems of classical physics and computational systems.

Let us first quickly review 'the solution' which some supporters of the computational approach take for granted. The basic idea is that each computational state corresponds to a physical state, and that a transition from one computational state to the next corresponds to the transition between the corresponding physical states. If this condition is satisfied, then (so they claim) a physical system realizes a computational one. Let us call the relation defined by the previous condition "emulation"³¹. Even though this relation is simple, and it seems promising, it will not work for one basic reason. We must recall first one important fact from computation theory. All universal computational systems are strongly irreversible³². This means that the system is able to perform two computations which start from different states, and which later produce the same state. Therefore, in general, given a computational state of a universal system, we cannot determine the preceding state. Also, it is not difficult to prove that a strongly irreversible system can only be emulated by a system which is strongly irreversible³³. The point now is that the mind presumably is a universal system, but we cannot take for granted that the concrete physical system which realizes the mind is strongly irreversible. Indeed, many systems of classical physics are

³¹ The reason for this name is that this is the same relation which holds when a computational system emulates a second computational system. Think, for example, of a universal Turing machine which emulates some other Turing machine. The total states of the emulated machine are mapped into the total states of the universal one, and each transition of the emulated machine corresponds to the (longer) transition between the corresponding states of the universal one. The formal analysis of the emulation relation in chapter 1, definition 4.

³² See ch. 1, th. 6, and the subsequent discussion.

³³ See theorem 6 of chapter 1 for the proof.

reversible, and we are assuming here that the brain is a classical system. Furthermore, even if the brain is strongly irreversible, this is an empirical fact, and an adequate theory of the relation between brain and mind should not depend on empirical contingencies³⁴. This argument alone is already sufficient for establishing that the emulation relation cannot be used to explain how the body can realize the mind. However, it is instructive to reach the same conclusion by means of a mental experiment.

Let us consider the way in which we would typically determine whether a concrete concrete physical system realizes a computational one. For definiteness, assume that the computational system is a Turing machine defined by a certain machine table, and that we want to decide whether a concrete piece of machinery realizes this Turing machine. In the first place, we look for three parts of the machinery, which correspond to the tape, to the head, and to the control unit of the Turing machine. The tape part must be a linear array of distinct addresses, and each address must be in one of a finite number of different states, which correspond to the symbols of the Turing machine. The head part must be a mechanism which is always located on exactly one tape address, has the capacity of changing the states of that address, and of moving to the address immediately to the left or to the right. Finally the state part (or control unit) must be in one of

³⁴ In general, this might not be true. However, the information processing approach claims that the relation between the brain and the mind is a special case of the realization relation between physical systems and computational systems. It seems obvious to require that a strongly irreversible system *may* be realized by a system which is not strongly irreversible. The emulation relation does not allow this possibility, for we can prove that no strongly irreversible system can be emulated by a system which is not strongly irreversible.

a finite number of mutually exclusive configurations, which correspond to the internal states of the Turing machine. In the second place, we check that whenever the head part is set (or reset) on address \mathbf{x} , the state part is set (or reset) to configuration \mathbf{q} , and the tape part is set (or reset) to configuration \mathbf{p} , the head part will then move to address \mathbf{x}' , the configuration of the state part will change to \mathbf{q}' , and the tape configuration will change to \mathbf{p}' , where \mathbf{x}' , \mathbf{q}' , and \mathbf{p}' correspond to the position, internal state, and tape configuration in which the Turing machine goes in one step when it is started in the position, internal state, and tape configuration which correspond to \mathbf{x} , \mathbf{q} , and \mathbf{p} . If all these conditions are satisfied, we conclude that this piece of machinery is a concrete realization of our Turing machine.

The interesting point is that, in the course of this procedure, we never checked whether or not a complete state of the Turing machine corresponds to a *physical state* of the apparatus, and whether or not a transition between complete computational states (of the Turing machine) corresponds to the transition between the corresponding physical states. In fact, we never reached the level of a physical state. What we checked is not a correspondence between the Turing machine and the physical states but, rather, whether a macroscopic description of the machinery's workings is true. Even though this mental experiment *does not prove* that the emulation relation is not adequate to describe the realization relation, it does seem to indicate that this relation must involve *true high level descriptions* of the physical system, *and not a direct mapping of computational*

states into physical states.

This is perhaps why Searle maintains that the realization relation between a physical system and a computational one necessarily involves an observer:

There is no way you could discover that something is intrinsically a digital computer because the characterization of it as a digital computer is always relative to an observer who assigns a syntactical interpretation to the purely physical features of the system. (Searle 1990b, 28)

Indeed, if a high level description is constitutive of the realization relation, it may seem that a subjective element is necessarily involved. For example, one might argue that there cannot be a description without an observer which does the describing. If you don't like this argument, you can certainly come up with a better one. But, before wasting your time, read below. It will soon be clear that descriptions are completely harmless, and that Searle's claim is plainly false.

That these descriptions cannot be merely subjective can immediately be seen from the fact that they must be *true*, something that Searle has completely overlooked. Searle says that "computational states are not *discovered within* the physics, they are *assigned* to the physics" (1990b, 27). Be that as it may, if these assignments lead to a computational description which is not satisfied by the system, then these assignments are obviously wrong. Therefore, the realization relation is not a pure matter of conventions. Perhaps Searle would then maintain that it can never be the case that some assignment turns out to be wrong, because any computational system can be realized by any physical system³⁵:

³⁵ From what Searle says it is not clear whether by "universal realizability" he means: [1] for any computational system there is a (sufficiently complex) physical system which realizes that computational system, or [2] any computational system can be realized by any (sufficiently big) physical system. The first part of clause 2. of the quotation in the text seems to assert [1], but the

On the standard textbook definition of computation,

1. For any object there is some description of that object such that under that description the object is a digital computer.
2. For any program there is some sufficiently complex object such that there is some description of the object under which it is implementing the program. Thus for example the wall behind my back is implementing the Wordstar program, because there is some pattern of molecule movements which is isomorphic with the formal structure of Wordstar. But if the wall is implementing Wordstar then if it is a big enough wall it is implementing any program, including any program implemented in the brain. (Searle 1990b, 26-27)

Contrary to what Searle affirms, these extraordinary claims do not follow from the standard definition of computation³⁶. In particular, this definition by no means implies that any computational system can be realized by any physical system³⁷. What we can prove is that *some* dynamical systems realize all computational systems. This is not just a trivial consequence of the definition of a computation. It is instead a rather involved theorem, and it is not at all clear whether this theorem applies to any *physical* system³⁸. Furthermore, the definition of the realization relation implies that there are systems which cannot realize all

example of the wall implies [2].

³⁶ It is far from obvious what Searle means by "standard textbook definition of computation". I take this standard definition to be the one you can find, say, in Davis (1958, ch. 1, def. 1.9). Obviously, Searle's extraordinary claims do not follow from this definition. If Searle has come across some alternative definition which implies these claims, he should let us know.

³⁷ This is exactly what Searle needs to justify his claim that a computational description of a physical system is arbitrary or conventional: "We wanted to know if there was not some sense in which brains were *intrinsically* digital computers in a way that green leaves intrinsically perform photosynthesis or hearts intrinsically pump blood. It is not a matter of us arbitrarily or "conventionally" assigning the word "pump" to hearts or "photosynthesis" to leaves. And what we were asking is, "Is there in that way a fact of the matter about brains that would make them digital computers? It does not answer that question to be told, yes, brains are digital computers because everything is a digital computer". (Searle 1990b, 26)

³⁸ See ch. 1, th. 10, and the subsequent discussion. This theorem only proves the existence of *some* dynamical system which realizes all computational systems, but it does not say anything about *physical* systems. Nevertheless, there may very well be physical systems which are computationally universal. On the other hand, it is trivial to prove that there are physical systems which are not computationally universal. Hint: think of a particle which is at rest.

computational systems.

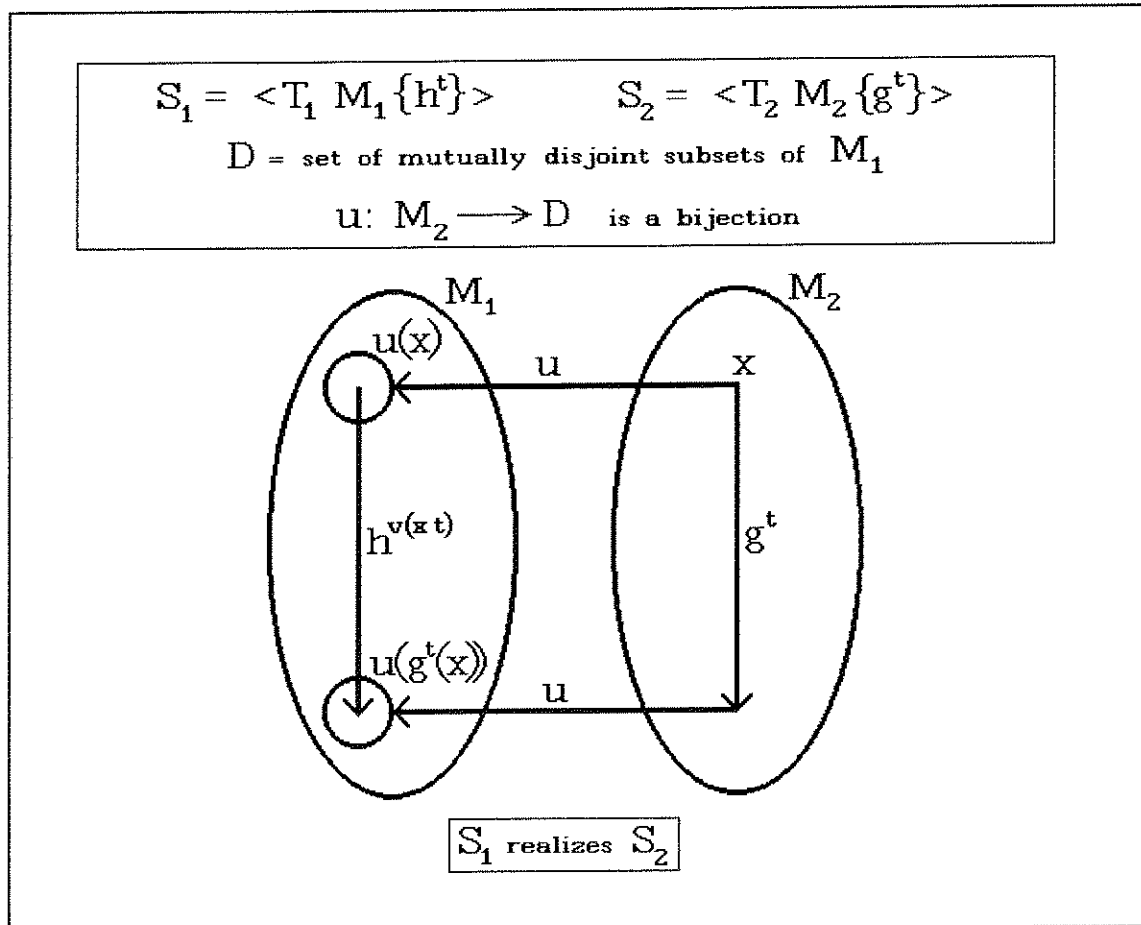


Figure 20: The realization relation

It is now time to explicitly say what the realization relation is³⁹. The basic idea is that system S_1 realizes system S_2 just in case (i) there is a bijection, u , between the states of S_2 and a set of disjoint *sets of states* of S_1 ; (ii) if $u(x)$ is the set of states which corresponds to x , and $u(y)$ is the set of states which corresponds to y , then the transition of S_2 from x to y corresponds to the transition of S_1 from x' to y' , where x' is an *arbitrary* state of S_1 which is in $u(x)$, and y' is a state of S_1 which is in $u(y)$.

³⁹ The formal analysis of the realization relation is in chapter 1, definitions 5 and 6. These two definitions are equivalent (see th. 7).

As you can immediately see, this relation does not involve any observer, and it is not trivially satisfied by any pair of systems S_1 and S_2 . It is also possible to prove that there are pairs of systems S_1 and S_2 such that S_1 realizes S_2 , S_1 is reversible, and S_2 is strongly irreversible. Therefore, the first argument which I have given will not work against this relation. Still, we might wonder that no description is constitutive of this relation, while our mental experiment suggests that the realization relation must involve descriptions. But this is not a real problem, for descriptions are present. These descriptions express the *sets of states* which correspond to the states of the realized system. Recall that any *state* of a dynamical system *completely describes* the system at some instant. Therefore, any set of states of a dynamical system can be thought as being expressed by an instantaneous description of that system. Conversely, any *instantaneous description* of a dynamical system always expresses a *set of states* of the system⁴⁰. It is for this reason that the presence of descriptions is completely harmless, and it does not provide any ground for a subjective interpretation of the realization relation. As John Winnie has suggested, this situation is analogous to the one we have when we apply probability theory to some concrete problem. Each event is assigned a probability, and events are formally defined as *sets* of elementary events. However, it is often useful to take a completely equivalent approach, which regards events to be expressed by descriptions. Descriptions are then assigned probabilities. Obviously, nothing

⁴⁰ A *complete* instantaneous description expresses a set which only contains one state. An *incomplete* instantaneous description expresses a set which contains at least two states.

changes if we switch from one formalism to the other. But, if we take Searle seriously, we should perhaps conclude that, whenever we use descriptions, we *ipso facto* accept a subjective interpretation of probabilities, and this is obviously absurd.

If it is still not clear how, in detail, sets of states (or instantaneous descriptions) are involved in our mental experiment, read again section 6 of chapter 1, and especially focus on the definition of a standard realization of a Turing machine, and on example 5.1. Our mental experiment can then be interpreted as follows. When we decide whether a concrete physical system realizes a Turing machine, we usually check whether that system satisfies the definition of a *standard realization* of a Turing machine. This definition essentially involves instantaneous descriptions of the system, which have the form: the head part of the system is set (or reset) on address \mathbf{x} , the configuration of the state part is set (or reset) to \mathbf{q} , and the configuration of the tape part is set (or reset) to \mathbf{p} . If we can empirically determine that the concrete physical system satisfies this definition, we can immediately conclude that it realizes the Turing machine. We do not need any further checking, because *we can prove that any standard realization of an arbitrary Turing machine realizes that Turing machine*. I have given this proof in example 5.1 of chapter 1. This result only applies to Turing machines. Nevertheless, given a specific *type* of computational system (register machines, systems specified by programs of some computer language, cellular automata, neural networks⁴¹, etc.)

⁴¹ Not all neural networks are computational systems, but those made of a finite number of units with discrete activation levels certainly are.

it is usually possible to define a standard realization in a natural way. We can then prove a theorem analogous to example 5.1 for this standard realization.

Finally, let me summarize the import of the previous discussion for the mind-body problem in the context of the information processing approach. This approach identifies the mind with a computational system concretely realized by the nervous system. We have seen that the nervous system can always be identified with a classical physical system. The problem of the relation between body and mind thus reduces to the problem of explaining how a dynamical system (nervous system) can realize, or implement, a second dynamical system (mind). The mathematical framework of dynamical system theory allows us to solve this problem in a completely general way. I have outlined this solution above, and I have developed the technical details in chapter one. Therefore, contrary to Searle's claims, the information processing theory can adequately explain how the mind is related to the underlying physical structure⁴².

⁴² As for the charge of anti-scientific thought (Searle 1990a, 31), the reader should judge for himself who the obscurantist is.

4. The problem of consciousness

The third question asks how the information processing theory of the mind can explain the subjective aspects of our mental life. How can we account for sensations, pains, dreams, hallucinations, emotions, feelings, desires, passions, if we have identified the mind with a computational system realized by a concrete physical system? First, let me introduce a bit of terminology. By a "*conscious state*" I mean *any state of subjective awareness, or any subjective experience or perception*. The third question can thus be rephrased in a more concise form: how can the information processing theory of the mind explain the nature of conscious states? Tim Maudlin (1989) has recently put forth an ingenious argument to show that this is impossible. His argument, however, heavily rests on the assumption that conscious states supervene on *computations of a Turing machine*. Therefore, even if we take Maudlin's argument to be sound and valid (which is controversial), it only shows that a *very specific* computational theory of consciousness is impossible. This argument cannot be applied to the approach which I am now going to outline⁴³.

The thesis I want to defend in this section is that the information processing

⁴³ Maudlin (1989, 412-13) claims that any computational theory of consciousness must accept two principles. If x is an arbitrary conscious state, and Δt is an arbitrary time interval, then: **(i)** there is a Turing machine C , and there is a computation c of this Turing machine such that, for any concrete physical system S , if S realizes C and S performs computation c during time interval Δt , then S supports conscious state x during time interval Δt ; **(ii)** for any concrete physical system S , if S supports conscious state x during time interval Δt , then there is a Turing machine C , and there is a non-trivial computation c of C such that the concrete physical system S realizes C and S performs computation c during time interval Δt . Contrary to Maudlin's claim, I will show below that it is possible to outline a computational theory of consciousness which is independent from these two principles. It thus follows that Maudlin's argument cannot be applied to this theory, for that argument is limited to theories which entail **(i)** and **(ii)**.

theory of the mind can *in principle* explain the *nature* of conscious states. Therefore, my discussion of this problem will be at a quite abstract level. My goal is not to propose a complete computational theory of consciousness. Rather, I am going to construct a *general framework* within which a detailed computational theory of consciousness *could* be developed⁴⁴.

The basic tenet of the information processing theory of the mind is that *the mind is a computational system realized by a concrete physical system*. We have seen in the previous section that this implies that *each mental state corresponds to a set of states of the concrete physical system*. On the other hand, *a conscious state can always be expressed by an instantaneous description of the mind*. For instance, if I have a sensation of red, this means that my mind satisfies the instantaneous description: *x has a sensation of red*. If I desire something, my mind satisfies the instantaneous description: *x desires something*. If I am in pain, my mind satisfies the instantaneous description: *x is in pain*. Therefore, since we have identified the mind with a computational system, any conscious state can be expressed by an instantaneous description of this system. But any computational system is a dynamical system, and an instantaneous description of a dynamical system always expresses a set of states of this system. It thus follows that *any conscious state is a set of mental states*. Furthermore, since any mental state in

⁴⁴ It should also be noticed that this framework is so general that it does not depend on the identification of the mind with a *computational* system realized by a concrete physical system but, rather, on its *consequence* that the mind is a *dynamical* system realized by a concrete physical system. Therefore if we only accept this weaker hypothesis, we may reject the information processing theory of the mind and still retain the theory of consciousness which I am going to sketch below.

this set corresponds to a set of physical states, *any conscious state corresponds to the union of all these sets of physical states*. Therefore, we can also conclude that *both mental states and conscious states correspond to sets of states of the concrete physical system which realizes the mind*. We thus see that the identification of the mind with a computational system realized by a concrete physical system entails that **(i)** mental states are states of a dynamical system realized by a concrete physical system; **(ii)** conscious states are sets of mental states; **(iii)** both mental states and conscious states correspond to sets of physical states.

Let me now make explicit two philosophical consequences of this view. In the first place, the theory I have just outlined is a special form of *functionalism*. In general, functionalists take the mind to be a system which is *realized* by a concrete physical system, but *cannot be identified with it*. Clearly, the identification of the mind with a *computational* system realized by a concrete physical system implies functionalism, for all computational systems are *discrete* dynamical systems, while physical systems are *continuous*. The denial of the possibility of a *complete* reduction of the mental to the physical is what distinguishes functionalism from the *identity theory*. This denial has the *methodological* consequence that cognitive science cannot be regarded as a special branch of physics.

In the second place, this *methodological* independence of the mental from the physical should not be confused with an *ontological* independence. Searle has claimed that the information processing theory of the mind rests on a kind of

dualism:

The polemical literature in AI usually contains attacks on something the authors call dualism, but what they fail to see is that they themselves display dualism in a strong form, for unless one accepts the idea that the mind is completely independent of the brain or of any other physically specific system, one could not possibly hope to create minds just by designing programs. (Searle 1990a, 31)

Frankly, what I fail to see is how a form of functionalism can be considered a dualistic position. The information processing theory of the mind maintains that the mind is a computational system *realized by a concrete physical system*. This obviously implies that *the mind is not ontologically independent from the concrete physical system which realizes it*, for a computational system cannot be a mind if it is not realized by some concrete physical system. Furthermore, *both mental states and conscious states cannot exist independently from a concrete physical system*, for mental states are states of a computational system *realized by a concrete physical system*, conscious states are sets of mental states, and both mental and conscious states thus *correspond* to sets of physical states.

Let me now consider a second type of criticism which might be raised against the theory of consciousness which I have outlined above. According to this theory, conscious states are sets of states of a computational system realized by a concrete physical system. Therefore, it is in principle possible to give a *complete account* of the nature of a specific conscious state *s* if we are able to specify: **(1)** the computational system which can be identified with the mind; **(2)** the concrete physical system which realizes the mind; **(3)** the specific set of mental states which is identical to the conscious state *s*; **(4)** the specific set of physical states which corresponds to the conscious state *s*. Suppose that we have been able to meet

all four conditions. Still, this account of the nature of the conscious state *s* cannot be *complete*, for none of these conditions involves a reference to the *introspectible qualitative character of s*. In other words, we cannot *identify* the conscious state *s* with a set of states of a computational system realized by a concrete physical system, for this identification leaves out one of the constitutive properties of *s*, namely *what it is like to be in conscious state s*. Therefore, this theory of the nature of conscious states is false⁴⁵.

Even though this argument has a strong intuitive appeal, it clearly rests on a false premise, namely that the introspectible qualitative character of a conscious state is one of its *properties*. *What it is like to be in conscious state s* is not a *property of s* but, rather, a *way of knowing s*. Therefore, this 'property' *must not be* part of an account of the nature of *s*. On the other hand, this 'property' *should be* part of a phenomenological (or first person) account of how conscious states are known by direct experience. But this is an *epistemological* question which the theory of consciousness I have outlined is *not* meant to address.

Finally, I wish to suggest a possible *scenario* which shows that a theory of conscious states constructed along the lines sketched above *might* be empirically tested. For definiteness, suppose that we have formulated a hypothesis **H** which consists of:

- (1) the specification of a computational system *M*;
- (2) the specification of a neurophysiological dynamical model *N* of some part of the nervous system *B* which realizes *M*;
- (3) the identification of a particular color sensation *s* with a specified set of states

⁴⁵ This argument is an adaptation from Nagel (1974). See in particular pp. 436-7 and 442-5.

- s of the system M ;
- (4) the specification of a set of states S of the system B such that S corresponds to s ⁴⁶;

then, the hypothesis H might be empirically tested by means of the following experiment:

suppose that we have some (*ethically acceptable*) means to force the part B of the nervous system of some subject in an arbitrary initial state x . Then, since the model N of B is known, we can determine the state y of B at a later time t . Also suppose that we instruct the subject to turn a switch *on* if he/she experiences the color sensation s , and to turn the switch *off* if he/she does not. Then, at time t , the switch should be *on* iff $y \in S$. Suppose we repeat this experiment for many initial states x and times t . Then, the hypothesis H is empirically confirmed just in case all the results are as expected.

⁴⁶ The specification of the set of states S which corresponds to the color sensation s must follow from (2) and (3). In fact, in the first place, S is equal to the union of all sets of states which correspond to the computational states in the set s specified by (3). In the second place, the set which corresponds to a computational state is determined by the realization relation specified by (2).

5. Is the mind a computational system?

The fourth question raises the doubt that, perhaps, we are not justified in taking the mind to be a computational system. There are many ways in which one could argue in favor of this view. I will start by reviewing Lucas' classic argument (1961, 1968a, 1968b). Penrose has recently advocated essentially the same position (1989, 108-12, 416-18). I will then propose a second argument which shows that the mind is not a computational system if it can be identified with a finite neural net whose units have continuous activation levels.

5.1 Lucas' argument

According to Lucas, we could establish that the mind is not a computational system on purely logical grounds. This would be a direct consequence of Gödel's incompleteness theorem for arithmetic. Gödel's theorem states that no axiomatizable theory which is consistent, and contains all the usual theorems of arithmetic⁴⁷, can prove all the arithmetical truths. The way Gödel reached this conclusion was by constructing an arithmetical truth which is not provable. This arithmetical truth is now known as the *Gödel sentence*. The Gödel sentence states its own unprovability so that, if G is a name for this sentence, the Gödel sentence simply is "G is not provable". Gödel's proof shows how, given any consistent, axiomatizable theory which contains the theorems of arithmetic, we can express G in the language of this theory. But then, in the first place, G must be

⁴⁷ Henceforth any such theory will be called "a formal theory of arithmetic".

true. Suppose it is false. Then, "G is not provable" is false, whence it is not the case that G is not provable. G is thus provable, so that G is a false consequence of the theory. But this is impossible, because logical consequence preserves truth, and the axioms of the theory are true. In the second place, G must be unprovable, for we know that "G is not provable" is true, whence G is not provable⁴⁸.

Now, so argues Lucas, this shows that the mind cannot be a computational system. Assume that the mind is a computational system. If the mind is a computational system, then it can be identified with a formal theory of arithmetic⁴⁹. Therefore, the mind can be identified with a formal theory of arithmetic. But then, the mind could never assert its Gödel sentence G, because the only way of asserting a sentence for a formal theory is proving that sentence and, by Gödel's theorem, this is impossible. On the other hand, the mind can

⁴⁸ This informal argument should not be confused with Gödel's proof. Nevertheless, thanks to Gödel, we know that there is nothing wrong with this argument, for we can always put it in a form which is logically impeccable.

⁴⁹ The argument which is usually given to support this second premise is the following. Suppose that the mind is a computational system. First, any computational system can be identified with an axiomatizable theory. Therefore, the mind is an *axiomatizable theory*. Second, we know that all the usual axioms of a formal theory of arithmetic are true, and we have the capacity of deducing all the consequences of these axioms. Therefore, from this and the first conclusion, the mind is an axiomatizable theory of *arithmetic*. Third, the set of all our beliefs is, at least in principle, consistent. Therefore, from this and the second conclusion, the mind is a *consistent* axiomatizable theory of arithmetic. It thus follows that, *if the mind is a computational system, then the mind is a consistent axiomatizable theory of arithmetic*.

Some people have questioned either the second or the third step of this argument. Some have noticed that the mind might not be a theory of *arithmetic*, for it might not contain all the theorems implied by the usual arithmetical axioms. Others have suggested that the mind might not be a *consistent* system. These objections, however, are not very strong, for they fail to show that *in principle* the mind cannot contain all the theorems of arithmetic, or it cannot be a consistent system.

assert G , because we can recognize that the Gödel sentence of *any* formal theory of arithmetic is true. Therefore, the mind is not a computational system.

This is a neat argument, and Penrose is probably right in implying that all the replies which have been proposed miss the point. Reviewing all these counters is beyond the scope of this chapter⁵⁰. I will instead discuss just one rebuttal which is particularly popular among cognitive scientists (Hofstadter 1980, 471-76). According to this view, it is simply false that, *for any formal theory of arithmetic*, we are able to *assert* its Gödel sentence G as an *arithmetical* sentence. The basic idea is this. Given the usual formal theories of arithmetic, we can certainly construct their Gödel sentences, and we are thus able to assert *arithmetical* sentences which are not provable by recognizing their truth. Nevertheless, if we considered an extremely complex theory of arithmetic, we could not construct the Gödel sentence for that theory⁵¹. So, even if we recognized its truth, we would not assert an unprovable arithmetical sentence. We would assert something else or, perhaps, nothing at all. Therefore, at that point, our mind would behave just like any other formal theory to which Gödel's theorem applies. In a few words: we

⁵⁰ For the interested reader, here is some bibliography. Putnam (1960), Smart (1961), Benacerraf (1967), Judson (1968), Good (1967, 1969), Lewis (1979), Lee Bowie (1982).

⁵¹ The argument goes like this. If we are given a first formal theory of arithmetic T_0 , we can construct its Gödel sentence G_0 . Then, we consider a second formal theory of arithmetic $T_1 = T_0 \cup \{G_0\}$ and we construct its Gödel sentence G_1 . We repeat this process so that we generate an infinite series $T_0, T_1, \dots, T_n, \dots$ of formal theories of arithmetic. The union of all these theories is still a theory of arithmetic, we thus formalize it, construct its Gödel sentence, and we then produce a second series $T_{\omega}, T_{\omega+1}, \dots, T_{\omega+n}, \dots$ of formal theories of arithmetic. We continue repeating this process. Then, for some extremely complex theory of arithmetic T_{α} (which is the union of an infinite series of formal theories of arithmetic) we will not be able to construct its Gödel sentence. Why? See the next footnote.

cannot outgödel Gödel in the long run.

This reply is extremely ingenious, but not completely satisfactory. It is perhaps true that, at some point, our capacity of constructing the Gödel sentence would break down. But perhaps not. If we understand Gödel's proof, we know how to construct this sentence for any formal theory of arithmetic, and it is not clear why we should not be able to carry out this construction for any particular case, however complex it may be. Granted, *in practice*, this might be impossible, but we need some ground for claiming that *this* is impossible *in principle*⁵². The defenders of this view have not provided any clear reason to establish this point, and this is why their argument sounds circular. In fact, there is one clear reason why we cannot outgödel Gödel: if we assume from the start that the mind is a formal system which satisfies the conditions of Gödel's theorem. But this is just begging the question.

Furthermore, even if we admit that in some specific case we cannot *construct* the Gödel sentence, this is irrelevant. We can still prove that, *for any* formal theory of arithmetic, *there is* a sentence which is true and cannot be proved by

⁵² The usual argument considers the series of formal theories of arithmetic which can be produced by adding to each previous theory its Gödel sentence. The problem comes when we try to put all these theories together in *one* formal theory. If the mind is a computational system, this process must break down at some point, because there is no effective procedure to reduce an arbitrary series of formal theories of arithmetic to one formal theory. Granted that this is the case, it is irrelevant. Lucas is not maintaining that our mind can carry out *this process* in any case. What he claims is that we can construct the Gödel sentence if we are *given an arbitrary* formal theory of arithmetic. If we can, then we can in principle outgödel any theory. And, in fact, we can produce the Gödel sentence for an arbitrary formal theory, however complex it may be. This follows from the fact that Gödel's proof is constructive.

that theory⁵³. Suppose that the mind is a computational system. Then, the mind can be identified with a formal theory of arithmetic T , and we can thus prove in T : *for any formal theory of arithmetic, there is a sentence which is true and cannot be proved by that theory*. Then, in particular, *this fact* applies to T , so that one of the theorems of T is: there is a sentence of T which is true and cannot be proved in T . The point now is that the predicate "true" is the truth predicate of T . Therefore, T contains its own truth predicate, which we know is impossible by Tarski's undefinability theorem⁵⁴. We have thus reached a contradiction. Therefore, the mind is not a computational system.

Nevertheless, the defenders of the computational approach should not be bothered too much by Lucas' argument (or by my more general version) for, at

⁵³ Gödel's theorem can be proved *without constructing the Gödel sentence* (Boolos and Jeffrey 1980, ch. 15). The precise statement of this form of the theorem is: for any theory which is an axiomatizable, consistent extension of Robinson's arithmetic, there is a sentence α such that neither α nor $\neg\alpha$ is provable in that theory. From this, since either α or $\neg\alpha$ must be true in the standard model of arithmetic, we obtain: for any theory which is an axiomatizable, consistent extension of Robinson's arithmetic, there is a sentence which is true but not provable in that theory. The point of this second argument is that *this semantic consequence of Gödel's theorem* cannot be proved in any formal theory of arithmetic. In fact, if it were provable in some formal theory of arithmetic T , also the following sentence would be provable in T : if T is an axiomatizable, consistent extension of Robinson's arithmetic, there is a sentence of T which is true but not provable in T . And this cannot be proved in T , because "true" is the truth predicate of T , and by Tarski's undefinability theorem, no axiomatizable consistent extension of Robinson's arithmetic contains its own truth predicate.

⁵⁴ A truth predicate of a formal theory of arithmetic T is any formula $\alpha(x)$ of T such that, for any formula β of T , ($\alpha(\ulcorner\beta\urcorner$) iff β) is provable in T , where " $\ulcorner\beta\urcorner$ " is the numeral which corresponds to the Gödel number of β . Tarski's undefinability theorem affirms that, for any formal theory of arithmetic T , no formula $\alpha(x)$ of T is a truth predicate of T . Intuitively, " $\ulcorner\beta\urcorner$ " is a name for β in T , and $\alpha(\ulcorner\beta\urcorner)$ can be interpreted as stating (" $\ulcorner\beta\urcorner$ is true") if, and only if, Tarski's definition of truth for that sentence, that is (" $\ulcorner\beta\urcorner$ is true iff β), is provable in T . Now, Tarski's undefinability theorem says that no predicate of a formal theory of arithmetic can be interpreted as stating the truth of an arbitrary formula of that theory. But we need exactly this predicate if we are to prove in T that there is a sentence of T which is true and cannot be proved in T . Tarski's undefinability theorem is an immediate consequence of the diagonal lemma (Boolos and Jeffrey 1980, ch. 15, lemma 2, exercise 15.1).

most, it shows that the mind is not a formal theory of arithmetic. It certainly does not show that the mind is not a computational system. This is an obvious point which, for some mysterious reason⁵⁵, has not been considered so far. Lucas, and even many of its critics, take for granted that, *if the mind is a computational system, then it can be identified with a formal theory of arithmetic*. But this is plainly false. To see this point, consider first that *any formal theory of arithmetic is not decidable*⁵⁶. This means that there is no effective procedure for determining whether an arbitrary sentence of that theory is a theorem. Second, *any computational system can instead be identified with a decidable theory*. In fact, any computational system can be identified with a discrete dynamical system (a cascade) $\langle T M H \rangle$ whose phase space M is decidable, and whose transition function H is Turing computable. This dynamical system can thus be identified with an axiomatizable theory. The axioms of this theory are all the possible states in the phase space M , and its inference rules are the transition function H and the identity function on M . Therefore, the set of all theorems of this theory is identical to the set of its axioms, that is, to the phase space M . Since M is decidable, *any computational system can thus be identified with a decidable theory*. This simple observation takes care of Lucas' argument, for it is just false that *if the mind is a*

⁵⁵ After all, the reason is not so mysterious. To see this point, we must take seriously the fact that computational systems are discrete dynamical systems (cascades) which can be effectively described. This fact has been overlooked so far.

⁵⁶ Any consistent extension of Robinson's arithmetic is not decidable (Boolos and Jeffrey 1980, ch. 15, th. 1). Since any formal theory of arithmetic is a consistent extension of Robinson's arithmetic, any such theory is not decidable.

*computational system, then it can be identified with a formal theory of arithmetic*⁵⁷.

5.2 Is the information processing theory of the mind consistent with the connectionist approach?

A different way of arguing that the mind is not computational considers the structure of the system with which the mind could plausibly be identified. According to the information processing theory of the mind, this system is a computational one. This means that there is an effective procedure for determining the possible states of the system, and that each transition from one mental state to another can always be described as an effective transformation of finite symbol structures. But is this hypothesis justified? The most recent developments in cognitive science do seem to indicate that it is not. A good way of arguing this point is reflecting on the following question: what if the mind could be directly identified with a network of nervous cells?

One of the reasons why this view was rejected in the sixties was that the cognitive abilities of neural networks, *as neural networks*, seemed extremely limited. For example no neural net of a simple kind can compute a basic logical

⁵⁷ More precisely, the second premise of Lucas' argument "if the mind is a computational system, then the mind is a formal theory of arithmetic" is false if the first premise "the mind is a computational system" is true. This follows from the fact that "the mind is a computational system" and "the mind is a formal theory of arithmetic" are inconsistent, for *any formal theory of arithmetic is not decidable*, while *any computational system can be identified with a decidable theory*.

operator, exclusive or⁵⁸. On the other hand, these limitations can easily be overcome if the neurons are connected in such a way to realize a *universal computer*. Since this seemed the only possible way of obtaining complex and flexible behavior from neurons, cognitive scientists concluded that *the mind should be identified with a universal computer which is realized by a network of nervous cells*.

It is however obvious that this view is justified *only if* the cognitive abilities of neural networks are in fact limited. But are they? Today we know much more on what neural networks of a certain complexity can do. For example, they can work very well as associative memories. We can store in a network many different 'packages of information' and, when the network is presented with a part of one of these packages, it will retrieve the missing parts⁵⁹. This might be what is going on when, starting from a clue, we are able to recall a whole lot of information which is associated with that clue. Notoriously, it is quite difficult to program a computer to handle this kind of task. Neural networks, on the other hand, do not need any programming. This remarkable ability is 'naturally produced' by some simple ways of linking the units together.

The point of this digression is just to make plausible that a neural net with the

⁵⁸ This simple kind of neural network is known as the "perceptron". The severe limitations of perceptrons were pointed out by Minski and Papert (1969). See also McClelland and Rumelhart (1988, 121-6).

⁵⁹ McClelland (1981) studied a simple interactive activation and competition network which displays this property. See also McClelland and Rumelhart (1988, ch. 2) and Rumelhart and McClelland (1986, vol. 1, 25-31).

right architecture might have all the cognitive abilities which we usually attribute to the mind. But then, if this is the case, it might be possible to *identify the mind with a network of nervous cells*. If this view were the correct one, could we still maintain that the mind is a *computational system*? The answer to this question would in general depend on the *nature* of the neural net with which the mind is identified.

Before considering this problem, we must eliminate certain simplistic ideas about the differences between neural networks and computational systems: (i) the processing of a neural net is parallel, while computational systems are serial; (ii) the representation and storing of information in neural nets is distributed, while in computational systems is local; (iii) a necessary feature of computational systems is the presence of a memory which contains complex symbolic structures. Each of these structures is analogous to a sentence of a formal language, for the whole structure and its parts can be interpreted. In particular, some of these structures explicitly represent rules or procedures, and a computational system always works by interpreting these rules and by applying them to other symbolic structures (data) stored in its memory. Neural networks, instead, do not have symbolic structures, and do not work according to rules.

The basic point is that the properties (i) - (iii) are not sufficient for distinguishing a neural network from a computational system. In fact, it is well known that there are computational systems which (a) are parallel; (b) store information in a distributed fashion; (c) do not have symbolic structures stored in their memory, and

do not work by interpreting explicitly represented rules. The classic example of such a computational system is a cellular automaton. A cellular automaton is constituted by a sequence of cells (or sites). Each cell always is in one of k possible states. The automaton updates synchronously the state of all its cells according to a rule which depends on the state of each cell, and on the states of its neighbors. For example, suppose that each cell has just two neighbors, the cell to its left and the cell to its right, and that the possible states are 0 and 1. Then, the following updating rule completely describes a specific cellular automaton: 000 \rightarrow 0, 001 \rightarrow 0, 010 \rightarrow 1, 011 \rightarrow 1, 100 \rightarrow 0, 101 \rightarrow 1, 110 \rightarrow 1, 111 \rightarrow 1.

Now, it is clear that any cellular automaton is a computational system, for the transition from one complete state to the next can be computed by a Turing machine, and the set of all possible states is decidable⁶⁰. Nevertheless, cellular automata process information in parallel, and the relevant information is always distributed across the whole arrangement of cells. In fact, it is never the case that the next automaton state is determined by the present state of *some* of the cells. Rather, *all* cells always contribute to the next state of the automaton. Also, for the same reason, a complete state of the automaton does not have separate parts, or

⁶⁰ More precisely, this is true for those cellular automata which satisfy (i) there is a special cell-state called the "quiescent state"; (ii) for any complete state of the automaton, at most a *finite number* of cells is not in the quiescent state; (iii) if a cell and all its neighbors are in the quiescent state, then the next state of this cell is again the quiescent state. I have proved in example 3.2 of chapter 1 that all linear cellular automata are computational systems.

symbolic structures, on which the automaton operates⁶¹. Finally, the rule which determines the behavior of the automaton is not explicitly represented as a part of its complete states, and the automaton does not work by interpreting this rule.

Conversely, we can reach the same conclusion by considering a special type of neural network. Suppose that each unit of a finite net has a finite number of possible activation levels. Then, the set of all complete states of the net can be identified with a finite set of finite strings of symbols, and any transition from one state to the next can obviously be computed by a Turing machine. Therefore, all networks of this type are computational systems, even though they usually process information in parallel, and they do not work by interpreting explicitly represented rules.

These two examples clearly show that the computational approach and the connectionist one are by no means exclusive. In fact, many connectionist networks are computational systems and, conversely, some computational systems like cellular automata are just a special type of connectionist network. 'Connectionism' and 'computationalism' are mutually exclusive only if computational systems are narrowly identified with mechanisms which somehow

⁶¹ One might argue that neighborhoods are these parts, and that the automaton works in parallel on all its neighborhoods. The point, however, is that neighborhoods usually overlap, so that the way a cellular automaton *actually* works is not correctly described as the application of rules to separate symbolic structures. On the other hand, such a description is possible, because any cellular automaton can be emulated by a Turing machine. This in fact shows that a cellular automaton *is* a computational system *in spite of* its non-standard way of working. It does not show that *it works* like a Turing machine. There are more computers in heaven and on earth than your Turing machine can dream of!

operate like a Turing machine⁶². But this view has no theoretical justification and, by focusing on superficial distinctions, completely misses the deeper point at issue.

It is now time to see what this deeper point is. First, we must recognize a simple fact: *any finite network whose units have continuous activation levels is not a computational system*. A complete state of any such net can always be identified with a finite sequence of real numbers and, since each unit has a continuous range of possible activation levels, the set of all possible complete states of this net is not denumerable. Therefore, this net cannot be isomorphic to a cascade whose set of possible states is decidable, for any decidable set is denumerable. It thus follows that any finite network with continuous activation levels is not a computational system⁶³. Now, the deep question is: *can the mind be identified with a finite neural net whose units have continuous activation levels?*

⁶² The important point is that *operating like a Turing machine* by no means is a necessary condition for being a computational system. On the other hand, *being describable as a Turing machine* certainly is. These two properties are essentially different, and they should never be confused. By "being describable as a Turing machine" I mean that the transition function of a system isomorphic to the system we are considering must be computable by a Turing machine, and that the phase space of this system is decidable (see ch. 1, def. 3).

⁶³ A computational system (see ch.1, def. 3) *could* be used to *approximate* the transitions of a network of this type. Nevertheless, if the real numbers involved are not computable, we cannot conclude that this approximation can be carried out to an arbitrary degree of precision. This is exactly the same situation that we have when we use computers to approximate the behavior of a physical system. A computational system cannot be *identified* with a physical system because physical systems are essentially more powerful than any computational system. This is why nobody, except perhaps Fredkin (Wright 1988), would confuse a computer simulation of a continuous system with the real thing. This is one of Searle's favorite points. Nevertheless, Searle is wrong in thinking that this identification is impossible because physical systems have some occult 'causal power' which computers lack. The difference is in the *power of processing information*: physical systems are analogic devices, so that they can change infinite amounts of information and, in general, they cannot be described in an effective manner. Computational systems, on the other hand, are limited to a finite amount of information, and they can always be effectively described.

If this identification is possible, then the mind is not a computational system. We do not yet know the answer to this question. Nevertheless, it is certainly reasonable to expect that this issue will ultimately be decided also, if not exclusively, on the basis of empirical investigations. To conclude, we can put this point in completely general terms. A deterministic dynamical system is *not* computational if: (i) it does not evolve in discrete time steps, or (ii) it is impossible to describe its phase space as a decidable set, or (iii) it is impossible to describe an arbitrary transition between two complete states as an effective transformation of finite symbol structures⁶⁴. Therefore, if the mind can be identified with a dynamical system which satisfies one of these conditions, the mind is not a computational system. The question of whether this identification is possible is an empirical one.

⁶⁴ These three conditions follow from the definition of a computational system (see ch. 1, def. 3). The precise formulation of these conditions is the following. A dynamical system S is not computational if (i) S is not a cascade or, for any cascade $S_1 = \langle T, M_1, H \rangle$ such that S is isomorphic to S_1 , (ii) the phase space M_1 is not decidable, or (iii) the transition function H is not Turing computable.

6. Cognitive science as a branch of dynamics

We now must consider what the conclusion of the previous section implies for our original question. This question is a *methodological* one⁶⁵. We want to know whether the hypothesis that the mind is a computational system realized by a concrete physical system should be one of the methodological foundations of cognitive science. By now it should be clear that it should not, for the possibility that the mind could be identified with a non-computational dynamical system is a real one, and it seems obvious that this possibility should be empirically investigated.

This conclusion leaves us in an uneasy situation. We thought that the information processing theory of the mind gave us an adequate picture of what the aims and methods of cognitive science are. We have instead discovered that this view cannot be the right one, for it rules out *a priori* a vast class of possibilities which instead should be empirically explored. My proposal for overcoming this methodological impasse is that we should seriously consider our fifth question: isn't it possible that cognitive science is better described by not assuming that *the mind* is its primary object, and by not assuming that this object necessarily is *computational*? The way in which I am going to answer this question is by explicitly exhibiting an alternative view which I believe to be superior to the

⁶⁵ An appropriate term also is "transcendental" in Kant's sense. We want to know whether the hypothesis that the mind is computational is one of the conditions which make cognitive science possible. The transcendental character of this hypothesis *is not* a merely philosophical interpretation. *This hypothesis is concretely used as a transcendental principle* by the defenders of the computational approach.

information processing theory of the mind. The basic idea of this methodological view is that the *primary* object of cognitive science is not the mind but, instead, a *particular set of dynamical systems*. I will call these systems "*cognitive systems*". A second name which is also appropriate is "*the mental*". I prefer the first term for it more clearly brings about the shift of perspective between this view and the one which primarily focuses on *the mind*.

Before attempting an explicit characterization of cognitive systems, I will spend a few words to explain why this view certainly deserves a try. One of the reasons which makes this perspective extremely appealing is that, if we are successful, we may demonstrate a structural identity between cognitive science and other natural sciences. For example, the object of physics is a particular set of systems whose generic properties can be described in detail. The same point also holds for some sciences which are concerned with the study of human behavior. For example, economics does not study *economic reality* (whatever that may be) but, instead, a set of systems which share some generic features. It is certainly reasonable to suppose that this general rule should also hold for cognitive science. Obviously, this analogical argument does not prove that cognitive science *can* be understood by focusing on the *generic properties of the systems which it studies*, but it does indicate that this view deserves to be taken seriously.

6.1 Cognitive systems

To understand what a cognitive system is, let us first go back to the information

processing theory of the mind and analyze in more details what it says about the entities which are the concrete focus of its empirical study. This view starts from the hypothesis that the mind is a computational system realized by a concrete physical system. This, however, is just the beginning of the story. What gives real bite to this approach is the further assumption that the mind is a *universal* computer of the usual kind. This essentially means that there is a working memory in which both programs and data can be stored, and that the mind has the capacity of executing *any* program which is stored in its memory. Depending on the software which actually is in the working memory, the mind is able to produce different behaviors. *Any different program in fact specifies a different computational system and, when the mind executes a particular program, it emulates the computational system specified by that program. This particular computational system has the cognitive properties or abilities which produce a behavior appropriate for a given situation, environment, or task.* For example, if the situation is playing chess, a chess program will be loaded into the working memory, and the mind will then execute it. Obviously, there may be environments for which no specific software is available. Presumably, however, there always are certain general purpose packages which can be executed in these contingencies. Some of these are learning routines, that is, programs which are able to create new programs, or improve old programs. Other general purpose packages are less smart. They simply produce 'defensive' or 'opportunistic' behavior: fleeing, refusal of engaging in a difficult task, research of a less demanding environment,

etc. Finally, certain situations may call for programs which establish goals, and then plan for the execution of other programs appropriate to reach these goals.

Three important points should be noticed. First, since the brain (or the nervous system) concretely realizes the mind, and the mind emulates a set of computational systems, all these systems are also realized by the brain⁶⁶. Second, the particular computational system which the mind emulates in a specific situation has specific cognitive abilities. Third, the aim of each empirical research is to describe in detail this particular computational system.

We thus see that, even within the framework of the information processing theory of the mind, *the object of each empirical research is not the mind itself but, instead, a specific computational system which the mind emulates in a specific situation. Furthermore, this computational system has particular cognitive abilities specific to that situation.* Therefore, one might be tempted to identify the set of all cognitive systems with the set of all computational systems which have cognitive abilities, and which the mind emulates while performing cognitive tasks. But, in this form, this idea will not work. First, what we need is a characterization of cognitive systems that *does not presuppose their being computational*, for some systems which have cognitive abilities are not computational. Second, *we do not want to presuppose the computational character of the mind*, for this is a question which is subject to empirical inquiry. Fortunately, there is a natural solution which satisfies both conditions. *This solution simply consists in identifying a cognitive*

⁶⁶ This follows from the transitivity of the realization relation, and from the fact that emulation is a special case of realization. See chapter 1, theorem 8, and corollary 8.1(a).

system with a dynamical system which is realized by a physical system, and has cognitive abilities⁶⁷. That is:

Definition 1 (cognitive systems)

π is a cognitive system iff:

- (1) π is a dynamical system;
- (2) there is a physical system σ such that σ realizes π ;
- (3) there is x such that x is a cognitive ability of π .

Let me explicitly say that this definition is not meant to give necessary and sufficient conditions which can be *indisputably* applied to each specific case. Part of the reason is that while the meaning of the first two conditions is sufficiently precise⁶⁸, the meaning of condition (3) is fuzzier. However, this need not be an irremediable defect of this definition. In fact, this only implies that we cannot establish once and for all which systems are cognitive and which are not. This determination instead depends on the specific case we are studying, and it might even include sociological or ideological factors. What we know is that, among all possible dynamical systems, *some* systems are relevant for cognitive science and that, as cognitive scientists, we should exclusively focus on those systems which

⁶⁷ Let me explicitly say that by a "cognitive system" I intend a *deterministic* cognitive system. I exclude *indeterministic* systems, not because I think that no stochastic system is cognitive but, rather, because I do not know how to precisely define, in a sufficiently general way, the class of the indeterministic dynamical systems.

⁶⁸ Condition (1) has a definite meaning, for " x is a dynamical system" is a set theoretical predicate. Condition (2) involves two predicates: " x is a physical system" and " x realizes y ". I have given a set theoretical definition of the realization relation in chapter 1 (see def. 6), and I take the meaning of " x is a physical system" to be sufficiently clear for our present purposes. What I intend is any system (concrete or abstract) which might be an object of physical study.

are realized by physical systems and have cognitive abilities. Obviously, we must give reasons for taking a specific dynamical system to have cognitive abilities. Nevertheless, establishing this fact is a genuine scientific discovery which is assessed by the cognitive community itself.

Furthermore, even if it is not possible to explicitly define the predicate " x is a cognitive ability of π ", we can nonetheless give a necessary condition. In fact, I am proposing to think of cognition in essentially dynamical terms. Therefore, it is quite natural to require that *the cognitive abilities of a system be a subset of its dynamical properties*. A dynamical property is, by definition, any property of a dynamical system which is shared by all the dynamical systems isomorphic to that system. That is, x is a dynamical property of π iff: π is a dynamical system, π has property x and, for any system $\pi^\#$ such that π is isomorphic to $\pi^\#$, $\pi^\#$ has property x . Therefore, if a system π has cognitive abilities, and its cognitive abilities are a subset of its dynamical properties, then π is a dynamical system, and any dynamical system which is isomorphic to it must have exactly the same cognitive abilities. Let me express this necessary condition on cognitive abilities in the form of a postulate:

Postulate 1 (cognitive abilities are dynamical properties)

If x is a cognitive ability of π , then x is a dynamical property of π ⁶⁹.

⁶⁹ This condition on cognitive abilities may seem too restrictive. It is in fact possible to think of a cognitive ability x of a dynamical system π_1 which does not exclusively depend on the dynamical structure of π_1 , but also on some further structure that the system has. Then, a second dynamical system π_2 which is isomorphic to π_1 , but does not have this further structure, would not

6.2 Cognitive studies as dynamical studies

My *methodological* proposal, then, consists in thinking of *cognitive science as a special branch of dynamics*. This means that the goal of each specific *cognitive study* is to produce a *dynamical explanation*⁷⁰ of a particular *cognitive system*. We have seen in chapter 3 that, in general, a dynamical explanation of a system or process consists in specifying a *dynamical model*⁷¹ of that system. Therefore, each specific cognitive study can also be thought as the attempt to *specify a dynamical model of a cognitive system*. Let me now make explicit the general picture of a cognitive study which this proposal implies.

According to this view, the first step of a cognitive study consists in *specifying a dynamical model* of a certain type. The particular form of this model, and the methods which lead to its specification, may be quite different from the forms and methods of other sciences, but there is an important identity in the underlying logical structure. For example, some of these dynamical models are computational, for they are specified by means of *computer programs*. Others consist of *neural networks*, which are specified by their *connections, weights, and*

have cognitive ability x . However, this situation can always be interpreted as follows. In the first place, x is not a cognitive ability of π_1 , for it is not one of its dynamical properties. In the second place, the cognitive ability of π_1 is instead the property $x^\#$ defined by: π has property $x^\#$ just in case there is a dynamical system $\pi^\#$ such that $\pi^\#$ is isomorphic to π and $\pi^\#$ has property x . The property $x^\#$ thus is a dynamical property of π_1 , for (i) π_1 is isomorphic to itself, and π_1 has property x ; (ii) for any system π_2 isomorphic to π_1 , there is a system isomorphic to it, namely π_1 , which has property x .

⁷⁰ See ch. 3, sec. 3 and, in particular, def. 7.

⁷¹ See ch. 3, sec. 2 and, in particular, def. 2 and def. 4.

by the *input-output characteristics* of each constitutive unit⁷². It can be shown that all these models are dynamical models in exactly the same sense as, for instance, mechanical models are⁷³. The main methodological difference between cognitive science and other natural sciences is that, at the moment, there is no *dynamical theory*⁷⁴ which subsumes all (most of) the models of this science. This might be a historical contingency, but it is also possible that, due to the particular character of cognitive models, a *highly general* dynamical theory is redundant or impractical. In other words, it *might* be possible that cognitive science can efficiently specify its dynamical models without explicitly formulating *theoretical principles*⁷⁵ of high generality.

The second step of a cognitive study consists in showing that the dynamical model we have specified is in fact a *model of the system which we want to explain*. In other words, we must produce a *justification* for the specific hypothesis

⁷² Some, *but not all*, connectionist networks are computational systems. It thus follows that some connectionist models are not computational. Other cognitive models which are not computational are those specified by systems of differential equations.

⁷³ For example, it is not difficult to show that EPAM (Feigenbaum's classic program which learns non-sense syllables, 1959, 1963) specifies a dynamical model. The proof of this fact, however, is long and it involves some technicalities. I plan to add the detailed analysis of this and other cases to an enlarged version of this dissertation. In general, cognitive models produce a temporal series of *outputs*. Each output can thus be thought as a value of the *observable magnitude(s)* of the model.

⁷⁴ See ch. 3, sec. 4 and, in particular, def. 10.

⁷⁵ Theoretical principles are hypotheses of a special form. See def. 8 of ch. 3 for the analysis of this form. Intuitively, a theoretical principle expresses the time evolution function of a *type* of magnitude as a mathematical function of the time evolution functions of other *types* of magnitude. As a typical example think of the second principle of dynamics: $\mathbf{A}(t) = \mathbf{F}(t) / \mathbf{M}(t)$, where $\mathbf{A}(t)$, $\mathbf{F}(t)$, and $\mathbf{M}(t)$ are, respectively, the time evolution functions of the acceleration, total force, and mass of an *arbitrary* body.

" P is a dynamical model of ψ ", where ψ is the system or process we study, and P is the dynamical model which we have specified. We have seen in chapter 3 (sec. 2.3) that, if ψ is a *concrete* system, the first step of this justification consists in showing that P is an *empirically adequate model* of ψ . This means that the time evolution laws of all the observable magnitudes of the model P must be consistent with all the (possible) measurements of those magnitudes. If this condition is satisfied, we may then be able to justify also the stronger hypothesis " P is a *dynamical model* of ψ ", for instance by producing further arguments which can be analyzed along the lines proposed by Giere (1985, 1988).

The third step of a cognitive study consists in showing that the concrete system ψ for which we have specified a dynamical model P is in fact a *cognitive system*. This means that we must justify the three hypotheses: (1) ψ is a dynamical system; (2) there is a physical system σ such that σ realizes ψ ; (3) there is x such that x is a cognitive ability of ψ . Hypothesis (1) follows from " P is a *dynamical model* of ψ "⁷⁶ so that, if this hypothesis is justified, then also (1) is. Hypothesis (2), on the other hand, follows from the fact that ψ is a *concrete* dynamical system⁷⁷. Therefore, we need only justify hypothesis (3). This can be obtained

⁷⁶ See ch. 3, def. 4. This definition states that P is a dynamical model of ψ just in case ψ is isomorphic to the dynamical system $\psi(P)$ generated by the model P . Since this isomorphism is an isomorphism *between dynamical systems*, this entails that ψ is a dynamical system.

⁷⁷ I take a *physical system* to be any system or process which might be an object of physical study. Therefore, some, *but not all*, physical systems are concrete systems. I take *concrete systems* to satisfy the following conditions. First, I assume that there are basic concrete systems, and that all basic concrete systems are physical systems. Second, a system ψ is concrete iff ψ is a basic concrete system, or there is some basic concrete system which realizes ψ . It thus follows that any concrete system is realized by some physical system. The analysis of the realization relation is in ch. 1, def. 5 and def. 6. This relation is reflexive and transitive.

by showing that the dynamical system $\psi(P)$ generated by the dynamical model P has some cognitive ability x . In fact, by postulate 1, the cognitive ability x is a dynamical property of $\psi(P)$. Furthermore, since P is a dynamical model of ψ , $\psi(P)$ is isomorphic to ψ , so that $\psi(P)$ and ψ must have the same dynamical properties. Therefore, if $\psi(P)$ has cognitive ability x , then also ψ must have this cognitive ability. A standard method for showing that $\psi(P)$ has cognitive abilities is by producing a computer *simulation* of the dynamical system $\psi(P)$, and by then using this simulation to perform cognitive tasks⁷⁸.

Finally, the fourth step of a cognitive study consists in *producing a detailed account* of the cognitive abilities which the system ψ has. In the first place, since ψ and the system $\psi(P)$ generated by the model P have the same cognitive abilities, this problem reduces to understanding the specific nature of the cognitive abilities of $\psi(P)$. In the second place, since cognitive abilities are properties of a *dynamical system*, a complete account of their nature is very likely to involve *the explicit use of methods and concepts from dynamical system theory*.

6.3 The symbolic, the connectionist, and the dynamical approach

Before concluding this section, let me briefly outline the relationships between my methodological proposal and the two main research programs in cognitive

⁷⁸ By a "*simulation* of a dynamical system" I intend a second system which *approximates*, but *does not* exactly reproduce, the state-space evolutions of the first system. This concept should thus be distinguished from both the *emulation* and *realization* relations, which instead express the idea of a system which is capable of *exactly reproducing* any possible evolution of another system. The emulation relation is a generalization of the concept of isomorphism, and it is a special case of the realization relation. See ch. 1, def. 2, def. 4, def. 5, and def. 6.

science -- the symbolic, and the connectionist approach. The symbolic approach (Newell and Simon, Pylyshyn, etc.) maintains that cognitive systems are included in a *proper subclass* of the computational systems. This subclass can be intuitively characterized as the class of all those computational systems which, somehow, *operate like* Turing machines. However, computational systems also include other dynamical systems which the symbolic approach has neglected: cellular automata, certain types of neural networks, etc. Perhaps, we could intuitively characterize this class of 'forgotten computational systems' as the class of parallel computers. What characterizes these systems is that they are not more powerful than Turing machines, but their way of functioning clearly differs (in some sense which can be made precise in each specific case) from Turing machines. *Some* of these non-standard computational systems are certainly relevant to the study of cognition, and should therefore be considered as possible objects of cognitive studies.

The connectionist approach identifies cognitive systems with a certain class of neural networks. This class intersects, but is not included in, the class of the computational systems. Computational neural networks include some of the computational systems 'forgotten' by the symbolic approach. All connectionist networks are dynamical systems but, with a few exceptions⁷⁹, the methodological implications of this fact have not been explicitly recognized.

Finally, the dynamical approach which I propose is characterized by three basic

⁷⁹ See, for example, Smolensky (1988).

features: (a) the explicit recognition of the dynamical character of cognition; (b) the disposition to apply methods and concepts from dynamical system theory to the study of 'the mental'; (c) the disposition to explore the *whole space* of the dynamical systems in order to locate and map the region of the *cognitive systems*. Therefore, this approach is the methodological framework which best allows for a plurality of empirical studies within a unified theoretical perspective.

7. Towards a dynamical theory of minds

In this final section, I will briefly discuss some implications of the dynamical approach for a *theory of minds*. I have proposed in the previous section to identify the object of cognitive science with the class of the cognitive systems, and we have seen that all cognitive systems have cognitive abilities. However, this condition is obviously *not sufficient* for a system to be a *mind*. I believe that *minds are a special type of cognitive systems which satisfy at least four more properties: (1) are realized by concrete physical systems; (2) have intentional states; (3) have conscious states; (4) are cognitively universal*⁸⁰. We should then ask whether a *dynamical theory of minds*⁸¹ can in principle explain these four properties. The thesis I want to defend is that it can, and that these explanations are at least as good as those provided by the information processing theory of the mind. In fact they are superior, for they do not *presuppose* the *computational* character of minds, but only the *weaker hypothesis* that minds are *dynamical* systems realized by concrete physical systems.

I take all four conditions which I have listed above to be uncontroversial for *any materialistic* theory of *minds*. Conditions **(2)**, **(3)**, and **(4)** are so general that they should follow from *any* theory of *minds*, *a fortiori* by a materialistic one. Condition **(1)** affirms that minds are systems realized by *concrete* physical systems, and this

⁸⁰ By "cognitive universality" I mean the property of having many and differentiated cognitive abilities, which allow a system to appropriately perform in a vast range of tasks or situations.

⁸¹ By a "*dynamical theory of minds*", I do not intend a dynamical theory in the technical sense which I have defined in ch. 3 (sec. 4.3, def. 11). Rather, I intend any conception of minds which accepts the hypothesis: *minds are dynamical systems realized by concrete physical systems*.

is just another way to express the basic materialistic intuition that minds cannot be independent from matter. I have formulated this condition in a very general way, so that it is consistent with both functionalism and the identity theory. In fact, functionalism can in general be characterized as stating **(1)** together with the further assumption that minds *are not* physical systems. The identity theory, on the other hand, affirms that minds *are* concrete physical systems, and this obviously entails **(1)**, for the realization relation is reflexive⁸².

Let us now see how a dynamical theory of minds can *in principle* explain these four properties. The basic assumption of this theory is that *minds are dynamical systems realized by concrete physical systems*. Therefore, condition **(1)** obviously follows from this hypothesis. Furthermore, we have seen (sec. 3) that the identification of the mind with a *dynamical* system allows us to explain in an objective and precise manner the *nature* of the relation between the mind and the underlying physical structure: as far as dynamical systems are concerned, a system S_1 realizes a second system S_2 just in case the states of S_2 correspond to sets of states of S_1 , and an arbitrary transition between two states of S_2 is mirrored by a transition between two states of S_1 which belong to the two sets which correspond to the two states of S_2 .

As for the *intentionality of mental states*, we have seen in sec. 2.3 how the information processing theory of the mind can in principle solve this problem:

⁸² I take *reflexivity* to be a basic condition which the realization relation between two systems of *any kind* must satisfy. A second basic condition is *transitivity*. I have proved that the realization relation *between dynamical systems* is reflexive and transitive (see. ch. 1, th. 8).

(i) the hypothesis that the mind is a computational system realized by a concrete physical system entails that all mental states have intensions; (ii) this hypothesis, together with a causal theory of reference⁸³, entails that (some) mental states have referential content. In fact, the second conclusion *does not* depend on the premise that the mind is a *computational* system, but only on the two weaker assumptions: (a) the mind is a *dynamical* system realized by a concrete physical system; (b) mental states have *parts or components*. Therefore, as far as the problem of the *referential content* of mental states is concerned, the solution which I have sketched in sec. 2.3 for the information processing theory of the mind is also valid for any *dynamical* theory of minds which entails (b).

We are thus left with the problem of the *intension* of mental states. We have seen (sec. 2.1) that the information processing theory of the mind can solve this problem because it takes the mind to be a *computational* system. All these systems can be identified with a special type of *theory*, and mental states turn out to be exactly the theorems of this theory. It thus follows that any mental state has an intension, namely the set of all (other) mental states which can be deduced from it⁸⁴. Now, if we are willing to slightly stretch the concept of a *theory*, we can generalize this solution to any *dynamical* system. From an abstract point of view, a theory is usually defined as a structure $\langle S T C \rangle$, where S is a non-empty set whose elements are the *sentences* of the theory, $T \subseteq S$ is the set of its *theorems*,

⁸³ See condition [2] of sec 2.3.

⁸⁴ Recall that I have proposed to identify the intension of a sentence (and in particular of a theorem) with the set of all sentences which follow from it.

and $C: P(S) \rightarrow P(S)$ is a function whose domain and codomain $P(S)$ is the set of all subsets of S . This function expresses the concept of *logical consequence*, so that it must have a certain number of properties which are satisfied by all the concrete concepts of logical consequence which the abstract definition is intended to describe. The most basic of these conditions is that the set T of all theorems be closed with respect to C , that is, if $C(T)$ is the set of all consequences of T , then $C(T) \subseteq T$. A second basic requirement is that any set of sentences be included in its consequence class, that is, $X \subseteq C(X)$ for any $X \subseteq S$. It thus follows that, for any theory, $T = C(T)$.

If we are willing to call a *theory* any structure $\langle S, T, C \rangle$ which satisfies these two basic conditions, then any dynamical system $\langle T, M, \{g^t\} \rangle$ generates a theory. In fact, take $S = T = M$ and, for any $X \subseteq S$, define $C(X) = \{y: g^t(x) = y \text{ for some } x \in X, \text{ and } t \in T\}$. Then, $C(T) \subseteq T$, and $X \subseteq C(X)$. We thus see that any *state* of a dynamical system is also a *theorem* of the theory generated by that system. Therefore, any state $z \in M$ has an *intension* which is equal to the consequence class $C(\{z\}) =$ the orbit of z . Finally, if we identify the mind with a dynamical system, any mental state is a state of this system, so that it has an intension which is equal to the orbit of that state in the mind's phase space.

We must now turn to *the property of having conscious states*. We have seen in sec. 4 that, according to the information processing theory of the mind, *conscious states are sets of mental states*, and that *they correspond to sets of physical states*. These conclusions do not in fact depend on the hypothesis that

the mind is a *computational* system realized by a concrete physical system, but only on the weaker premise that the mind is a *dynamical* system realized by a concrete physical system. Therefore, they are also valid for any *dynamical* theory of minds. This premise however, *does not* entail that minds *have* conscious states⁸⁵. It only implies that, *if they do*, the *nature* of conscious states is not mysterious, for they are just sets of mental states, that is, subsets of the phase space of a dynamical system. In other words, an *arbitrary* dynamical theory of minds can explain the *nature* of conscious states, but not their *existence*⁸⁶.

We have still to consider the property of *cognitive universality*. As it is well known, the information processing theory of the mind can provide a very good explanation of this property. So far, no competitor has even come close to match this explanation, and this is perhaps the strongest reason for thinking of the mind in computational terms. The proponents of this view take the mind to be a computational system very similar to a universal computer of the usual kind. This means that there is a *working memory* in which both programs and data can be stored, and that the mind has the capacity of executing *any* program which is stored in its memory. Depending on the program which actually is in the working memory, the mind is able to produce different behaviors. It is thus clear how the

⁸⁵ The same holds for the stronger premise: minds are *computational* system realized by concrete physical systems.

⁸⁶ On the other hand, it is obvious that *some* dynamical theories of minds may also explain the existence of conscious states. This follows from the fact that the property of having conscious states is consistent with the hypothesis: minds are dynamical systems realized by concrete physical systems.

property of cognitive universality is to be explained. We have only to further assume that *the mind is equipped with a number of different programs*, each of them appropriate for performing a specific task. Depending on the particular environment or situation, the appropriate program is loaded into the working memory. Then the mind executes it, thus producing the behavior required by that situation.

We thus see that, according to this explanation, *cognitive* universality is a consequence of *computational* universality. The classic example of a computationally universal system is a universal Turing machine. This machine is specified in such a way that, when a description of a second Turing machine is written on its tape, and a description of the initial complete state of this second machine is also written on the tape, the universal machine will produce in k steps a tape which contains both the description of the second Turing machine and the description of its next complete state. Some authors believe that computational universality can only be achieved by mechanisms which are similar to a universal Turing machine. In particular, they believe that *the presence of a memory which can be divided in at least two parts is necessary*. One part serves to represent the emulated mechanism (this part is the program), the other part represents the states through which the emulated mechanism evolves (Newell 1980, 148). The important point to bear in mind, however, is that this is just *one way* in which computational universality can be achieved. A system is computationally universal just in case it is able to *emulate* all computational systems. This means that for

each computational system, there is an injective mapping of its states into the states of the universal system such that an arbitrary transition between two states corresponds to a transition between the corresponding states of the universal system. This *does not imply* that the states of the universal system have two or more parts which somehow function as the parts of the tape of a universal Turing machine. Furthermore, since the emulation relation may hold between two *arbitrary* dynamical systems, there are systems which *are not computational* and, nonetheless, *emulate all computational systems*⁸⁷.

We should finally recognize an important general fact. The property of universality *is not* limited to *computational* systems, but it is instead shared by a much wider class of *dynamical* systems. Intuitively, a dynamical system is universal if it is able to *exactly reproduce* the behavior of any system in a given class. We have seen (ch. 1, def. 4, def. 5, and def. 6) that both the *emulation* and the *realization* relations formally express the idea of a system which is capable of *exactly reproducing* any possible evolution of another system. Therefore, given a class of dynamical systems X , we can define two concepts of universal system:

⁸⁷ A simple way of obtaining one of these systems is by constructing a system with continuous time which 'mirrors' a universal Turing machine. Let $S_1 = \langle T_1, M_1, G \rangle$ be an arbitrary dynamical system with discrete time (a cascade). Then, there is a dynamical system $S_0 = \langle T_0, M_0, \{g^t\} \rangle$ such that: (i) $T_0 =$ the real numbers, if $T_1 =$ integers, or $T_0 =$ the non-negative real numbers, if $T_1 =$ the non-negative integers; (ii) S_0 emulates S_1 . **Proof:** choose $M_0 = \{ \langle w, x \rangle \text{ such that } 0 \leq w < 1 \text{ and } x \in M_1 \}$. For any $t \in T_0$, let $\text{floor}(t)$ be the integer obtained by eliminating the decimal part of t . Let G^k be obtained by iterating k times the transition function G of S_1 , and let $G^0 =$ identity function on M_1 . Define $g^t(w, x) = \langle (w + t - \text{floor}(t)), G^{\text{floor}(w+t)}(x) \rangle$. Then, by construction, S_0 is a dynamical system, and S_0 emulates S_1 (hint: identify x with $\langle 0, x \rangle$, and G with g^1) q.e.d. If, in particular, S_1 is a universal Turing machine, then S_0 emulates all computational systems, for the emulation relation is transitive. However, S_0 is not a computational system, for S_0 has not discrete time, and its phase space M_0 is not denumerable.

a system is E-universal with respect to X just in case it *emulates* all systems in class X; a system is R-universal with respect to X just in case it *realizes* all systems in class X.

The important fact about these concepts is that *they are not limited to computational systems, but they instead apply to arbitrary dynamical systems*. It thus clear how a *dynamical* theory of minds can explain the property of *cognitive universality*: we can just think of this property as a special case of either *E-universality* or *R-universality*. In other words, a system S is cognitively universal if there is a sufficiently broad class of cognitive systems C such that S is E-universal with respect to C or S is R-universal with respect to C⁸⁸.

Finally, let me summarize the main results of this section. The information processing approach has thought of the mind as a universal *computer*, partly because this assumption has seemed necessary for explaining the unrestricted or *universal* abilities of the mind. Contrary to this view, I have shown that a *dynamical* theory can provide an explanation at least as good: cognitive universality can be understood as a special case of two very general *dynamical properties*⁸⁹: *E-universality* or *R-universality*. Furthermore, a dynamical theory of minds can also explain the nature of conscious states, the intentionality of mental

⁸⁸ It should be noticed that this is an explanation of the *nature* of cognitive universality, not of its *existence*. If we are interested in this second problem, then an *evolutionary* explanation suggests itself: cognitive universality has been selected for because systems with this property have a high fitness.

⁸⁹ Since isomorphism is a special case of either the emulation or the realization relation, and these relations are transitive, any system isomorphic to a E-universal (or R-universal) system is also E-universal (or R-universal). Therefore, both E-universality and R-universality are dynamical properties.

states, and the nature of the relation between the mind and the underlying physical structure. The hypotheses on which these explanations are based are *weaker* than the corresponding hypotheses of the information processing theory. On the other hand, the accounts that they provide are *at least as good*. Therefore, if one of the goals of cognitive science is the construction of a theory of minds, then the dynamical approach is the most promising one.

Conclusions

In the penultimate section of chapter 4, I have proposed a new view of the aims and methods of cognitive science - *the dynamical approach*. According to this methodological theory, the object of cognitive science is the set of all *cognitive systems*, and the goal of a cognitive study is to produce a *dynamical explanation* of a specific cognitive system. The investigations of the previous chapters have provided the conceptual framework for answering three basic questions which concern this view: (i) What is a cognitive system? (ii) Is the nature of all cognitive systems computational? (iii) What, exactly, is a dynamical explanation of a cognitive system?

The answer which I have proposed for the first question is that a cognitive system is a dynamical system which is realized by a physical system and has cognitive abilities (ch. 4, def. 1). The analysis of the realization relation (ch. 1, def. 5 and def. 6) gives a definite content to this answer. I have also proposed to think of cognitive abilities as a subset of the dynamical properties of a cognitive system (ch. 4, postulate 1), and I have then suggested that a cognitive study should produce a *detailed account* of the cognitive abilities of the specific system it attempts to explain. Since cognitive abilities are properties of a *dynamical system*, a detailed account of their nature is likely to involve the *explicit use of methods and concepts from dynamical system theory*.

The analysis of the concept of a computational system (ch. 1, def. 3, and ch. 2, def. 8) implies that the second question has a negative answer: *there are cognitive systems which are not computational*. This is clear once we realize that any *continuous* dynamical system¹ is not computational, and that many connectionist networks which have cognitive abilities are continuous dynamical systems. I believe that other types of continuous systems² may have cognitive abilities as well. In the last section of chapter 4, I have proposed to think of minds as a *special type of cognitive systems* which satisfy at least four further properties: **(1)** are realized by *concrete* physical systems; **(2)** have intentional states; **(3)** have conscious states; **(4)** are cognitively universal. The question of whether all minds are *computational* systems will ultimately be decided by empirical considerations, but my guess is that an affirmative answer is quite unlikely.

Finally, since all cognitive systems are dynamical systems, the answer to the third question is provided by the analysis of a *dynamical explanation* which I have developed in chapter 3 (def. 7). According to this view, a dynamical explanation consists in specifying a *dynamical model* (ch. 3, def. 2 and def. 4) of the system or process which we want to explain. An important consequence of this view is that it does not limit *a-priori* the form of the dynamical models which can be considered by cognitive science. For example, some of these models are

¹ By a "*continuous* dynamical system" I mean a dynamical system $\langle T, M, \{g\} \rangle$ such that at least one of the following conditions is satisfied: (i) the time T of the system is the set of the real numbers (or the set of the non-negative reals); (ii) the phase space M is not denumerable.

² For instance, systems specified by differential equations.

computational, for they are specified by means of computer programs. Others consist of *neural networks*, which are specified by their connections, weights, and by the input-output characteristics of each unit. Finally, a third type of models are those specified by *systems of differential equations*. It thus follows that the *dynamical approach* which I propose is the methodological framework which best allows for a plurality of empirical studies within a unified theoretical perspective.

Bibliography

Agassi, Joseph

1965 The nature of scientific problems and their roots in metaphysics. In *The critical approach to science and philosophy*, 189-211. See Bunge 1965.

Anderson, John R.

1990 *Cognitive psychology and its implications*. New York: W. H. Freeman.

Arbib, Michael A.

1964 *Brains, machines, and mathematics*. New York: McGraw-Hill.

Arnold, V. I.

1977 *Ordinary differential equations*. Cambridge MA: The MIT Press.

Baltzer, Wolfgang, C. Ulises Moulines, and Joseph D. Sneed

1987 *An architectonic for science: The structuralist program*. Dordrecht: D. Reidel Publishing Co.

Barwise, J., H. J. Keisler, and K. Kunen

1980 *The Kleene symposium*. Amsterdam: North Holland.

Beckman, Frank S.

1980 *Mathematical foundations of programming*. Reading MA: Addison Wesley.

Belnap, Neul D. Jr., and Thomas B. Jr. Steel

1976 *The logic of questions and answers*. New Haven: Yale Univ. Press.

Benacerraff, Paul

1967 God, the devil, and Gödel. *The Monist* 51:9-32.

Bennett, Charles H.

1973 Logical reversibility of computation. *IBM J. Res. Develop.* November:525-32.

1982 The thermodynamics of computation - a review. *International Journal of Theoretical Physics* 21, 12:905-40.

Bennett, Charles H., and Rolf Landauer

1985 The fundamental physical limits of computation. *Scientific American* 253, July:48-56.

- Berlekamp, E. R., J. H. Conway, and P. Y. Guy
 1982 *Winning ways for your mathematical plays*. London: Academic Press.
- Blum, Lenore, Mike Shub, and Steve Smale
 1989 On a theory of computation and complexity over the real numbers: NP-Completeness, recursive functions and universal machines. *Bulletin of the American Mathematical Society* 21, 1:1-46.
- Boden, Margareth A.
 1988 *Computer models of mind*. Cambridge: Cambridge Univ. Press.
- Boolos, George S., and Richard C. Jeffrey
 1985 *Computability and logic*. Cambridge: Cambridge Univ. Press.
- Braithwaite, R. B.
 1964 *Scientific explanation: A study of the function of theory, probability and law in science*. Cambridge: At the University Press.
- Bromberger, Sylvain
 1966 Why-questions. In *Mind and Cosmos*, 86-111. See Colodny 1966.
- Bunge, Mario, ed.
 1965 *The critical approach to science and philosophy*. New York: Free Press.
- Burks, Arthur W.
 1970 *Essays on cellular automata*. Urbana: University of Illinois Press.
- Churchland, Paul M.
 1979 *Scientific realism and the plasticity of mind*. Cambridge: Cambridge Univ. Press.
 1988 *Matter and consciousness*. Revised edition. Cambridge MA: The MIT Press.
 1989 *A neurocomputational perspective*. Cambridge MA: The MIT Press.
- Churchland, Paul M., and Patricia Smith Churchland
 1990 Could a machine think? *Scientific American* 262, 1: 32-37.
- Churchland, Paul M., and Clifford A. Hooker, eds.
 1985 *Images of science*. Chicago: The Univ. of Chicago Press
- Colodny, Robert G., ed.
 1966 *Mind and Cosmos*. Pittsburgh: Pittsburgh Univ. Press.
 1972 *Paradigms and paradoxes: The philosophical challenge of the quantum domain*. Pittsburgh: Univ. of Pittsburgh Press.

- Dalla Chiara, Maria Luisa, and Giuliano Toraldo di Francia
 1981 *Le teorie fisiche: Un'analisi formale*. Torino: Boringhieri.
- Davis, Martin
 1958 *Computability and unsolvability*. New York: McGraw-Hill.
 1974 *Computability*. New York: Courant Institute of Mathematical Sciences.
- Davis, Martin, ed.
 1965 *The undecidable*. Hewlett NY: Raven Press.
- Deutsch, D.
 1985 Quantum theory, the Church-Turing principle, and the universal quantum computer". *Proc. R. Soc. Lond. A* 400: 97-117.
- Devaney, R. L.
 1989 *An introduction to chaotic dynamical systems*. Menlo Park: Benjamin Cummings.
- Dewdney, A. K.
 1989 Two-dimensional Turing machines and Tur-mites. *Scientific American* 261, 9:180-83.
 1991 Insectoids invade a field of robots. *Scientific American* July:118-121
- Drake, Stillman
 1989 *History of free fall*. Toronto: Wall & Thompson.
- Dreyfus, Hubert L., and Stuart E. Dreyfus
 1986 *Mind over machine*. New York: The Free Press.
- Ellis, Brian
 1985 *What science aims to do*. In *Images of Science*, 48-74. See Churchland and Hooker 1985.
- Farmer, Doyne, Tommaso Toffoli, and Stephen Wolfram, eds.
 1984 *Cellular automata*. Amsterdam: North Holland Physics Publishing.
- Feigenbaum, Edward A.
 1959 *An information processing theory of verbal learning*. P-1817. Santa Monica CA: The Rand Corporation Mathematics Division.
 1963 The simulation of verbal learning behavior, in *Computers and thought*. See Feigenbaum and Feldman 1963.

- Feigenbaum, Edward A., and Julian Feldman, eds.
1963 *Computers and thought*. New York: Mc-Graw Hill.
- Fetzer, J., ed.
1988 *Aspects of artificial intelligence*. Kluwer Academic Publishers.
- Feyerabend, Paul
1975 *Against method*. London: Verso.
- Fodor, Jerry A.
1975 *The language of thought*. New York: Thomas Y. Crowell.
1979 *The modularity of mind*. Cambridge MA: The MIT Press.
- Fodor, Jerry A., and Zenon W. Pylyshyn
1988 Connectionism and cognitive architecture: A critical analysis. *Cognition* 28:3-71.
- Francois, Robert
1986 *Discrete iterations*. Berlin: Springer Verlag.
- Friedman, Harvey
1971 Algorithmic procedures, generalized Turing algorithms, and elementary recursion theory. In *Logic colloquium '69*, 361-89. See Gandy and Yates, 1971.
- Galilei, Galileo
1914 *Dialogues concerning two new sciences*. Translated by Henry Crew and Alfonso de Salvio. Reprint. New York: Dover Publications, Inc.
1953 *Dialogue concerning the two chief world systems*. Translated by Stillman Drake. Berkeley: University of California Press.
1957 *Discoveries and Opinions of Galileo*. Translated by Stillman Drake. Garden City NY: Doubleday & Company, Inc.
1979 *Galileo's notes on motion*. Edited by Stillman Drake. Firenze: Istituto e Museo di Storia della Scienza.
- Gandy, R. O.
1980 Church's thesis and principles for mechanisms. In *The Kleene symposium*, 123-48. See Barwise, Keisler, and Kunen, 1980.
- Gandy, R. O., and C. M. E. Yates, eds.
1971 *Logic colloquium '69*. Amsterdam: North Holland Publ. Co.
- Garey, Michael R., and David S. Johnson
1979 *Computers and intractability*. San Francisco: W. H. Freeman and Co.

- Giere, Ronald N.
 1984 *Understanding scientific reasoning*. Second edition. New York: Holt, Rinehart and Winston.
 1985 Constructive realism. In *Images of Science*, 75-98. See Churchland and Hooker 1985.
 1988 *Explaining Science: A cognitive approach*. Chicago: The University of Chicago Press.
- Giunti, Marco
 1988 Hattiangadi's theory of scientific problems and the structure of standard epistemologies. *The British Journal for the Philosophy of Science* 39:421-39.
- Good, I. J.
 1967 Human and machine logic. *The British Journal for the Philosophy of Science* 18:144-47.
 1969 Gödel theorem is a red herring. *The British Journal for the Philosophy of Science* 19:357-58.
- Harnad, Stevan
 1987 Category induction and representation. In *Categorical perception: The groundwork of cognition*. See Harnad (ed.) 1987.
 1989 Minds, machines, and Searle. *J. Expt. Theor. Artif. Intell.* 1:5-25.
 1990 The Symbol Grounding Problem. *Physica D* 42:335-46.
- Harnad, Stevan, ed.
 1987 *Categorical perception: The groundwork of cognition*. New York: Cambridge Univ. Press.
- Haugeland, John, ed.
 1987 *Mind design*. Cambridge MA: The MIT Press.
- Harrington, L. A., & al., eds.
 1985 *Harvey Friedman's research on the foundations of mathematics*. Amsterdam: North Holland Publ. Co.
- Hattiangadi, J. N.
 1978 The structure of problems (part I). *Philosophy of the Social Sciences* 8:345-65.
 1979 The structure of problems (part II). *Philosophy of the Social Sciences* 9:49-76.

- Hempel, Carl G.
 1965 *Aspects of scientific explanation and other essays in the philosophy of science*. New York: The Free Press.
- Herken, Rolf, ed.
 1988 *The universal Turing machine: A half century survey*. Oxford: Oxford Univ. Press.
- Heyting, A., ed.
 1959 *Constructivity in mathematics*. Amsterdam: North Holland.
- Hofstadter, Douglas R.
 1979 *Gödel, Escher, Bach*. New York: Basic Books Inc.
- Hofstadter, Douglas R., and Daniel C. Dennett, eds.
 1981 *The mind's I*. New York: Basic Books Inc.
- Hopcroft, John E., and Jeffrey D. Ullman
 1979 *Introduction to automata theory, languages, and computation*. Reading MA: Addison Wesley.
- Hsu, C. S.
 1987 *Cell-to-cell mapping*. New York: Springer Verlag.
- Johnson-Laird, Philip N.
 1988 *The computer and the mind*. Cambridge MA: Harvard Univ. Press.
- Judson, Webb
 1968 Metamathematics and the philosophy of mind. *Philosophy of Science* 35:156-178
- Kalmar, Laszlo
 1959 An argument against the plausibility of Church's thesis. In *Constructivity in mathematics*, 72-80. See Heyting, 1959.
- Kitcher, Philip, and Wesley C. Salmon, eds.
 1989 *Scientific Explanation*. Vol. XIII of *Minnesota studies in the philosophy of science*. Minneapolis: University of Minnesota Press.
- Kleene, S. C.
 1967 Computability. In *Philosophy of science today*, 36-45. See Morgenbesser 1967.

- Koertge, Noretta
1982 Explaining scientific discovery. In *PSA 1982*, vol. 1.
- Kocak, Huseyin
1986 *Differential and difference equations through computer experiments*. New York: Springer Verlag.
- Korb, Kevin
1991 Searle's AI program. *Journal of Theoretical and Experimental Artificial Intelligence* 2:283-96.
- Kuhn, Thomas S.
1970 *The structure of scientific revolutions*. Second edition. Chicago: The University of Chicago Press.
- Langley, Pat, Herbert A. Simon, Gary L. Bradshaw, and Jan M. Zytkow
1987 *Scientific discovery: Computational explorations of the creative process*. Cambridge MA: The MIT Press.
- Lakatos, Imre
1970 Falsification and the methodology of scientific research programmes. In *Criticism and the growth of knowledge*. See Lakatos and Musgrave 1970.
1976 *Proofs and refutations*. Cambridge: Cambridge Univ. Press.
- Lakatos, Imre and Musgrave, Alan, eds.
1970 *Criticism and the growth of knowledge*. Cambridge: Cambridge Univ. Press.
- Laudan, Larry
1977 *Progress and its problems: Towards a theory of scientific growth*. Berkeley: University of California Press.
- Lee Bowie, G.
1982 Luca's number is finally up. *Journal of Philosophical Logic* 1982:279-85.
- Lewis, David
1979 Lucas against mechanism II. *Canadian Journal of Philosophy* IX, 3:373-376.
- Lucas, J. R.
1961 Minds machines and Gödel. *Philosophy* 36:112-27.
1968a Satan stultified: A rejoinder to Paul Benacerraff. *The Monist* 52:145-158.
1968b Human and machine logic: A rejoinder. *British Journal for the Philosophy of Science* 19:155-56.

- Margolus, Norman
 1984 Physics-like models of computation. In *Cellular automata*, 81-95. See Farmer, Toffoli, and Wolfram, 1984.
- Martin, Olivier, Andrew M. Odlyzko, Stephen Wolfram
 1984 Algebraic properties of cellular automata. *Comm. Math. Phys.* 93:210-58.
- Maudlin, Tim
 1989 Computation and consciousness. *The Journal of Philosophy* LXXXVI, 8:407-32.
- McClelland, James L.
 1981 Retrieving general and specific information from stored knowledge of specifics. *Proceedings of the Third Annual Meeting of the Cognitive Science Society*, 170-172.
- McClelland, James L., and David E. Rumelhart
 1988 *Explorations in parallel distributed processing: A handbook of models, programs, and exercises*. Cambridge MA: The MIT Press.
- Minsky, Marvin
 1967 *Computation: Finite and infinite machines*. Englewood Cliff NJ: Prentice Hall.
- Minsky, M., and S. Papert
 1969 *Perceptrons*. Cambridge MA: The MIT Press.
- Morgenbesser, Sidney, ed.
 1967 *Philosophy of science today*. New York: Basic Books.
- Newell, Allen, and Herbert Simon
 1972 *Human problem solving*. Englewood Cliffs NJ: Prentice Hall.
- Nagel, Ernest
 1961 *The structure of science: Problems in the logic of scientific explanation*. New York: Harcourt, Brace & World, Inc.
- Penrose, Roger
 1989 *The emperor's new mind*. Oxford: Oxford Univ. Press.
- Péter, Rózsa
 1981 *Recursive functions in computer theory*. Chichister: Ellis Horwood.

- Polya, G.
 1945 *How to solve it: A new aspect of mathematical method*. Princeton: Princeton Univ. Press.
 1962 *Mathematical discovery*. Vol . 1. New York: John Wiley & Sons.
 1965 *Mathematical discovery*. Vol . 2. New York: John Wiley & Sons.
- Popper, Karl Raimund
 1963 *Conjectures and refutations*. London: Routledge and Kegan Paul.
 1972 *Objective knowledge: An evolutionary approach*. Oxford: Clarendon Press.
- Pour-El, Marian B., and J. Ian Richards
 1989 *Computability in analysis and physics*. Berlin: Springer Verlag.
- Przelecki, Marian
 1969 *The logic of empirical theories*. London: Routledge & Kegan Paul.
- Putnam, Hilary
 1960 Review of Nagel E. and Newman R. Gödel's Proof. *Philosophy of Science*, 27:205-7.
- Pylyshyn, Zenon W.
 1980 Computation and cognition: Issues in the foundations of cognitive science. *The Behavioral and Brain Sciences* 3:111-69.
 1984 *Computation and cognition*. Cambridge MA: The Mit Press.
- Pylyshyn, Zenon W., and Liam J. Bannon, eds.
 1989 *Perspectives on the computer revolution*. Norwood NJ: Ablex Publishing Corporation.
- Pylyshyn, Zenon W., and William Demopoulos, eds.
 1986 *Meaning and cognitive structure*. Norwood NJ: Ablex Publishing Corporation.
- Rapaport, W.
 1988 Syntactic semantics. In *Aspects of artificial intelligence*. See Fetzer 1988.
- Rogers, Hartley Jr.
 1987 *Theory of recursive functions and effective computability*. Cambridge MA: The MIT Press.
- Rose, H. E., and John C. Shepherdson, eds.
 1975 *Logic colloquium '73*. Amsterdam: North Holland Publ. Co.

- Rumelhart, David E., and James L. McClelland, eds.
 1986 *Parallel distributed processing*. 2 vols. Cambridge MA: The MIT Press.
- Salmon, Wesley C.
 1984 *Scientific explanation and the causal structure of the world*. Princeton NJ: Princeton University Press.
 1990 *Four decades of scientific explanation*. Minneapolis: University of Minnesota Press.
- Searle, John
 1980 Minds, brains, and programs. *The Behavioral and Brain Sciences* 3:417-24.
 1984 *Minds, brains and science*. Cambridge MA: Harvard Univ. Press.
 1990a Is the brain's mind a computer program? *Scientific American* 262, 1:26-31.
 1990b Is the brain a digital computer? *Proceedings and Addresses of The American Philosophical Association* 64, 3:21-37.
- Shank, Roger C., and Kenneth Mark Colby, eds.
 1973 *Computers models of thought and language*. San Francisco: W. H. Freeman and Co.
- Shepherdson, John C.
 1975 Computation over abstract structures: serial and parallel procedures and Friedman's effective definitional schemes. In *Logic colloquium '73*, 445-513. See Rose and Shepherdson, 1975.
 1985 Algorithmic procedures, generalized Turing algorithms, and elementary recursion theory. In *Harvey Friedman's research on the foundations of mathematics*, 285-308. See Harrington & al., 1985.
 1988 Mechanisms for computing over abstract structures. In *The universal Turing machine: A half century survey*, 581-601. See Herken 1988.
- Smart, J. J. C.
 1961 Gödel's theorem, Church's theorem, and mechanism. *Synthese* 13, 1961:105-10.
- Smith, Alvy Ray III
 1971 Simple computation-universal cellular spaces. *Journal of the Association for Computing Machinery* 18, 3:339-53.
- Smolensky, P.
 1988 On the proper treatment of connectionism. *The Behavioral and Brain Sciences* 11:1-74.

- Sneed, Joseph D.
 1971 *The logical structure of mathematical physics*. Dordrecht: D. Reidel Publishing Co.
- Stegmüller, Wolfgang
 1976 *The structure and dynamics of theories*. New York: Springer-Verlag.
 1979 *The structuralist view of theories*. Berlin: Springer-Verlag.
- Suppe, Frederick
 1989 *The semantic conception of theories and scientific realism*. Urbana: University of Illinois Press.
- Suppe, Frederick, ed.
 1974 *The structure of scientific theories*. Urbana: University of Illinois Press.
- Suppes, Patrick
 1957 *Introduction to Logic*. Princeton NJ: D. Van Nostrand Company, Inc.
 1967 What is a scientific theory? in *Philosophy of Science today*, 55-67. See Morgenbesser, 1967.
 1969 *Studies in the methodology and foundations of science: Selected papers from 1951 to 1969*. Dordrecht: D. Reidel Publishing Company.
- Szlenk, Wieslaw
 1984 *An introduction to the theory of smooth dynamical systems*. Chichester: John Wiley & Sons.
- Turing, Alan M.
 1956 Computing machinery and thinking. *Mind* LIX, 236:433-60.
 1965 On computable numbers, with an application to the entscheidungsproblem. In *The undecidable*, 115-54. See Davis, 1965.
- Tye, Michael
 1989 *The metaphysics of mind*. Cambridge: Cambridge Univ. Press.
- van Fraassen, Bas C.
 1967 Meaning relations among predicates. *Nous* 1:161-179.
 1969 Meaning relations and modalities. *Nous* 3:155-67.
 1970 On the extension of Beth's semantics of physical theories. *Philosophy of Science* 37:325-39.
 1972 A formal approach to the philosophy of science. In *Paradigms and paradoxes*, 303-366. See Colodny 1972.
 1980 *The scientific image*. Oxford: Clarendon Press.
 1985 Empiricism in philosophy of science. In *Images of science*, 245-308. See Churchland and Hooker 1985.

- 1989 *Laws and symmetry*. Oxford: Clarendon Press.
- van Gelder, Tim
- 1991 Connectionism and Dynamical Explanation. *Proceedings of the 13th Annual Conference of the Cognitive Science Society*. Hillsdale NJ: L. Erlbaum Associates. 499-503.
- 1992 The proper treatment of cognition. To appear in *Proceedings of the 14th Annual Conference of the Cognitive Science Society*.
- Veloso, Paulo A. S.
- 1984 Aspectos de uma teoria geral de problemas. *Cadernos de Historia e Filosofia da Ciência* 7:21-42.
- Vichniac, Gerard Y.
- 1984 Simulating physics with cellular automata. In *Cellular automata*, 96-116. See Farmer, Toffoli, and Wolfram, 1984.
- Wagner, Klaus, and Gerd Wechsung
- 1986 *Computational complexity*. Dordrecht: D. Reidel Publishing Co.
- Wainwright, Robert T.
- 1974 Life is universal! *Winter Simulation Conference* January 14-16:449-59.
- Wessels, Linda
- 1974 Laws and meaning postulates. In *PSA 1974*, 215-234.
- Westfall, Richard S.
- 1977 *The construction of modern science: Mechanisms and mechanics*. Cambridge: Cambridge University Press.
- Winograd, Terry
- 1983 *Language as a cognitive process*. Vol. I, *Syntax*. Reading MA: Addison-Wesley.
- Wolfram, Stephen
- 1983a Statistical mechanics of cellular automata. *Reviews of Modern Physics* 55, 3:601-44.
- 1983b Cellular automata. *Los Alamos Science* 9:2-21.
- 1984a Computer software in science and mathematics. *Scientific American* 56:188-203
- 1984b Cellular automata as models of complexity. *Nature* 311:419-24.
- 1984c Universality and complexity in cellular automata. In *Cellular automata*, 1-35. See Farmer, Toffoli, and Wolfram, 1984.
- 1984d Computation theory of cellular automata. *Comm. Math. Phys.* 96:15-57.

1986 Random sequence generation by cellular automata. *Advances in Applied Mathematics* 7:123-69.

Wolfram, Stephen, ed.

1986 *Theory and applications of cellular automata*. Singapore: World Scientific.

Wright, Robert

1988 *Three scientists and their gods*. New York: Times Books.

Marco Giunti

Qualifications

- 1981 Laurea in Filosofia, Dept. of Philosophy, Univ. of Florence, Italy.
1981/82 Scuola di Perfezionamento in Filosofia, Univ. of Florence, Italy.
1988 M.A. in Philosophy of Science, Dept. of History and Philosophy of Science, Indiana University.
1992 Ph.D. in Philosophy of Science, Dept. of History and Philosophy of Science, Indiana University.

Teaching

- 1985/87 Scientific Reasoning, X200, Dept. of History and Philosophy of Science, Indiana University.
1986/87 Logical Theory II, P506, grader, Dept. of Philosophy, Indiana University.
1987/89 Minds, Brains, and Computers, X355-X755, assistant instructor, Dept. of History and Philosophy of Science, Indiana University.

Fellowships and assistantships

- 1985 (a) Fellowship from the Italian Ministry of Education to attend the Graduate Program in History and Philosophy of Science at Indiana University.
(b) Fellowship from Indiana University Graduate School.
1985/86 Fellowship from Indiana University Foundation.
1989/90 Research assistantship from Dept. of History and Philosophy of Science.

Publications

- 1983 Popper and Lakatos: due diverse giustificazioni del falsificazionismo. *Dimensioni*, 28-29:128-56.
1988 Hattiangadi's Theory of Scientific Problems and the structure of standard epistemologies. *The British Journal for the Philosophy of Science* 39, 4:421-39.

Associations

- Member of the Philosophy of Science Association.
Member of the American Philosophical Association.
Member of the Italian Philosophical Society (Società Filosofica Italiana).