Formal Logic
Classical Problems and Proofs

Luis M. Augusto

ISBN 978-1-84890-317-3

College Publications
Scientific Director: Dov Gabbay
Managing Director: Jane Spurr

http://www.collegepublications.co.uk

Contents

Preface xv

I Formal Logic: Form, Meaning, and Consequences 1

1 Preliminary notions 3
1.1 Formal languages: Alphabets and grammars 3
1.2 Logical languages: Form and meaning 7
1.2.1 Object languages and metalanguages 7
1.2.2 Logical sentences: From categorical propositions to set-theoretical expressions 8
1.3 Logic and metalogic: Proofs and metaproofs 11
1.3.1 Induction, mathematical and structural 12
1.3.2 Proof by contradiction 13
1.4 Logic and computation: Turing machines, decidability and tractability 15

2 Logical form 33
2.1 Logical languages and well-formed formulas 33
2.1.1 Alphabets, expressions, and formulas logical 33
2.1.2 Orders 35
2.2 Formalizing natural language 40
2.3 Argument form 46
2.4 Normal forms and substitutions for L1 52
2.4.1 Literals and clauses 52
2.4.2 Negation normal form 53
2.4.3 Prenex normal form 54
2.4.4 Skolem normal form 54
2.4.5 Conjunctive and disjunctive normal forms 57
2.4.6 Substitutions and unification for L1 62

3 Logical meaning 77
3.1 Truth: Values, tables, and functions 77
3.2 The Boolean foundations of logical bivalence 79
3.3 Compositionality and truth-functionality 83

3.4 Classical interpretations and valuations 88
3.5 Meaning and form 93

4 Logical consequences 99
4.1 Logical consequence: A central notion 99
4.1.1 Consequences and systems logical, inferential, and deductive 99
4.1.2 Syntactical consequence and proof theory 105
4.1.3 Semantical consequence and model theory 111
4.1.4 Adequateness of a deductive system 116
4.2 Logical theories and decidability 120
4.2.1 Theories, subtheories, and extensions 121
4.2.2 FOL theories and decidability 123
4.2.2.1 Finite satisfiability and ground extensions 125
4.2.2.2 Finite models and prefix classes 132

II The System CL and the Logic CL 139

5 The language of classical logic 141
5.1 Some preliminary remarks 141
5.2 L1 and classical subsets/extensions thereof 144
5.2.1 The classical connectives 144
5.2.2 The quantifiers of CFOL 148
5.3 Applications of L1 148
5.3.1 Arguments: Categorical syllogisms 148
5.3.1.1 Evaluating arguments with Euler diagrams 150
5.3.1.2 Evaluating arguments with Venn diagrams 151
5.3.2 Logic programming (I) 153
5.3.2.1 The language (of) Prolog 155
5.3.2.2 Increased expressiveness and ambivalent syntax 158
5.3.2.3 Programs and substitutions 159
5.3.3 Logic design: Logic circuits 162

6 Classical logical consequence 175
6.1 Classical ♡-consequences 175
6.1.1 Classical syntactical ♡-consequences 176
6.1.2 Classical semantical ♡-consequences 178
6.2 Classical ♦-consequences 180

7 CL and extensions 183
7.1 The logic CL 183

7.2 The extension $CL^=$: CL with equality 186

8 Classical FO theories and the adequateness of CFOL 193

III Classical Models 201

9 Three formal semantics for classical logic 203
9.1 Tarskian semantics 204
9.2 Herbrand semantics 206
9.3 Algebraic semantics: Boolean algebras 212

IV Classical Proofs I: Direct Proofs 221

10 The validity problem, or *VAL* 223
10.1 The *Entscheidungsproblem* and Turing's negative answer . 223
10.2 *VAL* and direct proofs 227
10.3 The complexity of *VAL* 231

11 Hilbert-style systems 237
11.1 The axiom system $\mathcal{L}$ 238
11.1.1 The propositional system $\mathcal{L}$ 238
11.1.2 The FO system $\mathcal{L}$ 245
11.2 Further Hilbert-style systems 246
11.2.1 The class $\mathscr{H}$ 246
11.2.2 Other systems 248

12 Gentzen systems 253
12.1 The natural deduction calculus $\mathcal{NK}$ 253
12.1.1 The propositional calculus $\mathcal{NK}$ 254
12.1.2 The FO predicate calculus $\mathcal{NK}$ 265
12.1.3 The extension $\mathcal{NK}^=$ for $CL^=$ 270
12.2 The sequent calculus $\mathcal{LK}$ 271

V Classical Proofs II: Indirect Proofs 285

13 The satisfiability problem, or *SAT* 287
13.1 *SAT* and refutation proofs 288
13.1.1 The different forms of *SAT* 288
13.1.2 Indirect proofs 292
13.2 The complexity of *SAT* 295

13.3 Herbrand's Theorem and the SAT 300

14 The resolution calculus 311
14.1 The resolution principle 313
14.1.1 The resolution principle for propositional logic . . 313
14.1.2 The resolution principle for FOL 317
14.2 Resolution refinements 324
14.2.1 Semantic resolution 325
14.2.2 Linear resolution: Logic programming (II) 332
14.3 Paramodulation 346

15 The analytic tableaux calculus 365
15.1 Analytic tableaux as a propositional calculus 367
15.2 Analytic tableaux as a FO predicate calculus 376
15.2.1 FOL tableaux without unification 378
15.2.2 FOL tableaux with unification 380

Bibliography 385

Bibliographical references 387

Index 395

List of Figures

1.1.1 A syntactic, or derivation, tree. 6
1.2.1 Relations of inclusion and exclusion in diagrammatic representation. Source: Venn (1881). (Work in the public domain.) 8
1.4.1 Computer model of a Turing machine. 17
1.4.2 A Turing machine that computes the function $f(n, m) = n + m$ for $n, m \in \mathbb{N}^{+}$. 19
1.4.3 The hierarchy of complexity classes with corresponding tractability status. 21
1.4.4 The encodings $\langle M_T \rangle$ and $\langle M_T, z \rangle$. 24
1.4.5 State diagram of a Turing machine. 30

2.2.1 Formalizations for English by means of the language of classical propositional logic. 45
2.2.2 Formalizations for English by means of the language of classical FO logic. 47
2.3.1 Some classical formally correct arguments. 51
2.3.2 Two invalid argument forms. 52
2.4.1 Tseitin transformations for the connectives of L. 61
2.4.2 Unifying the pair $\langle P(a, x, h(g(z))), P(z, h(y), h(y)) \rangle$ . . 67
2.4.3 A FOL argument. 73

3.3.1 A truth table with $2^3 = 8$ rows. 84
3.3.2 Truth tables for the connective $\rightarrow$ in the 3-valued logics L_3, K_3^W, and Rn_3. 87
3.5.1 The properties of a Boolean algebra. 95

4.1.1 The complete lattice $\mathcal{S} = \left(2^A, \subseteq\right)$ for $A = \{a, b, c\}$. 103
4.1.2 Adequateness of a deductive system $\mathrm{L} = (\mathsf{L}, \Vdash)$. 119

5.1.1 Venn diagram of the set A. 143
5.2.1 Diagrammatic representations of the connectives of O_{L}. . . 145
5.2.2 Diagrammatic representations of the logical connectives $O_G = \left\{\uparrow^2, \downarrow^2, \nleftrightarrow^2\right\}$. 147
5.2.3 Euler diagrams for the classical quantifiers. 149
5.3.1 Euler diagrams of an invalid (1) and a valid (2) argument. 152

5.3.2 Venn diagram with eight minterms. 153
5.3.3 A Venn-diagram representation of argument A1. 154
5.3.4 A Venn-diagram representation of argument A2. 154
5.3.5 From a binary switch (i) to a series-parallel connection (v). 164
5.3.6 Logic gates and their graphical representations. 165
5.3.7 A logic circuit for the function $f(x_1, x_2, x_3) = (x_1 \wedge x_2) \vee (x_1 \wedge x_3)$. 166
5.3.8 A logic circuit for $f(x_1, x_2, x_3) = x_1 \wedge (x_2 \vee x_3)$. 166
5.3.11 Properties of XOR. 167
5.3.9 Two functionally equivalent logic circuits. 168
5.3.10 The De Morgan's laws and the NOR and NAND gates. . . 169
5.3.12 Logic circuits. 174

11.1.1 Proof of $\vdash_{\mathfrak{L}} \phi \rightarrow \phi$ 239
11.1.2 Proof of an argument in $\mathfrak{L}p$ 241
11.1.3 Proof in $\mathfrak{L}q$ of a valid syllogism. 246

12.1.1 A proof of a propositional derivation in $\mathcal{NK}$. 256
12.1.2 A proof in $\mathcal{NK}$ of the distributivity property for $\wedge$. 259
12.1.3 Proof of $\vdash_{\mathcal{NK}} ((P \rightarrow Q) \wedge (P \rightarrow R)) \rightarrow (P \rightarrow (Q \wedge R))$. . . 260
12.1.4 Proof of an argument in (extended) $\mathcal{NK}$. 261
12.1.5 Proof of $\vdash_{\mathcal{NK}} \phi \leftrightarrow \neg\neg\phi$. 263
12.1.6 A proof with universal generalization. 266
12.1.7 An example of universal instantiation. 267
12.1.8 An example of existential generalization. 267
12.1.9 An example of existential instantiation. 268
12.1.10 A FO $\mathcal{NK}$ proof. 269
12.1.11 A proof in $\mathcal{NK}^{=}$. 271
12.2.1 Proof in $\mathcal{LK}$ of axiom $\mathscr{L}2$ of the axiom system $\mathscr{L}$. 276
12.2.2 Proof in $\mathcal{LK}$ of a FO validity. 277

13.2.1 A tableau for the Turing machine M. 298
13.3.1 Closed semantic tree of $C = \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4, \mathcal{C}_5\}$ in Example 13.31. 303
13.3.2 A closed semantic tree. 304

14.1.1 A refutation tree. 315
14.1.2 A propositional argument as input in Prover9/Mace4. . . . 316
14.1.3 Output by Prover9: A valid propositional argument. . . . 316
14.1.4 Output by Prover 9: A valid formula. 318
14.1.5 Output by Mace4: A counter-model. 318
14.1.6 A resolution-refutation failure tree. 320
14.1.7 Input in Prover9/Mace4: A FO theory. 320

14.1.8 Output by Prover9. 321
14.1.9 Output of Prover9: A valid FO argument. 322
14.2.1 A PI-resolution tree. 328
14.2.2 Hyper-resolution of $\Xi = (\mathcal{C}_3; \mathcal{C}_1, \mathcal{C}_2)$. 331
14.2.3 A hyper-resolution deduction tree. 331
14.2.4 Theory of distributive lattices and commutativity of meet: Input in Prover9/Mace4. 332
14.2.5 Proof by Prover9 of the commutativity of meet in a distributive lattice. 333
14.2.6 A linear-resolution refutation tree. 334
14.2.7 Trace by SWI-Prolog. 338
14.2.8 A failed proof tree. 340
14.2.9 A successful reduction interpreted as a resolution proof. . . 342
14.2.10 A LI-resolution proof tree. 343
14.2.11 A SLD-resolution proof. 346
14.2.12 A complete proof tree. 347
14.2.13 SWI-Prolog answering a query and outputting traces for some "true" instantiations. 348
14.2.14 SWI-Prolog traces of a "true" and a "false" instantiation. . 349
14.3.1 Theory of commutative groups: Input in Prover9/Mace4. . 353
14.3.2 Output by Prover9. 354

15.1.1 Analytic tableaux expansion rules: $\alpha\beta$-classification. . . . 369
15.1.2 A closed propositional tableau. 373
15.2.1 Analytic tableaux expansion rules: $\gamma\delta$-classification. 377
15.2.2 A closed FO tableau without unification. 379
15.2.3 A closed FO tableau with unification. 382

List of Algorithms

2.1 PNF transformation. 55
2.2 Skolemization. 56
2.3 Tseitin transformation. 60
2.4 The Robinson algorithm. 65

14.1 Binary resolution. 312
14.2 Reduction. 340

15.1 Analytic tableaux proof. 366

Preface

Often spoken of as the science of reasoning, *logic* can be *formal* or *informal*. While it is not unequivocal–there is significant overlap between both–, the use of these two adjectives allows us to distinguish between a largely mathematical from a substantially psychological approach, respectively, to logic. This might appear unwarranted to those well-acquainted with logic as an object language, but at the metalanguage and/or metalogical levels it becomes clear that formal logic has its foundations in mathematics, namely in what can be called abstract mathematics, whereas informal logic reposes on psychological theories of human reasoning. This book is an introduction to formal logic.

A second major distinction in contemporary logic segregates *classical logic* from the *non-classical logics*. These–note the plural–are typically rivals of the former–note the singular–, it being meant by this that they aim at replacing it in many contexts and/or applications. This rivalry notwithstanding, they are either extensions or restrictions of classical logic, which means that anyone advocating a non-classical logic should be well-versed in classical logic. This book is an introduction to classical logic.

While formal classical logic is certainly interesting per se, today its study is often associated to computer science with a plethora of computational implementations in view. This association of logic and computation can be roughly captured by the expression *computational logic*. This book is an introduction to computational logic.

Do we then need to specify that this book is an introduction to formal classical computational logic? Not really, because in it we take the adjectives *formal* and *computational* to be so intimately related that they can be often considered synonyms. This synonymy is more typically to be found between the expressions *formal language* and *computer language*, but we discuss here the language of classical logic as first and foremost a formal language, and hence the redundancy of the adjective *computational* in the title.

This book is thus an introduction to formal classical logic with its contemporary uses in mind, to wit, *logical problems* that are in fact *decision problems* that are in fact *computational problems* whose *proofs* are delegated to computer software. In effect, logic is—arguably—all about proving, but proofs can be costly, often impossibly so, in terms of space and time, it being meant by this that proofs require storage space (i.e., a physical memory) and they take time to be computed; hence, monetary costs are also often associated to proofs, as space and time, as well as human work, cost money. Given these costs, unrealistic for human computers and undesirable for companies, today most proofs are delegated to (partly) automatic provers, namely the so-called *SAT solvers*. These are software based on the (Boolean) satisfiability problem, or *SAT*. This is the dual of the (Boolean) validity problem, or *VAL*, at the core of the conception of the digital computer via Hilbert's *Entscheidungsproblem* and the Universal Turing Machine.

These two problems, *VAL* and *SAT*, can be said to be the two classical problems that initiated the computational history of formal classical logic, a history that can be more immediately traced back to the *Entscheidungsproblem*, but that actually also requires digressions into the work of the likes of J. Venn, G. Frege, and A. Turing–if not Aristotle, too. In particular, we discuss the classical formal semantics conceived by, or originating in the work of, G. Boole, J. Herbrand, and A. Tarski. While this, as said, is an introduction to formal classical logic, we dispense with the adjective "classical" between "formal" and "logic" in the title, because this book has as its backbone these two semantical problems. The fragment "Formal logic: Classical problems" indicates that our introduction to formal logic is so via the classical problems, first and foremost *VAL* and *SAT*, but then also all the decision and computational problems that can be formulated in terms of these, namely with computer implementations in mind.

But, as stated above, logic is–arguably–all about proving. Without (adequate) proof systems at hand, these two problems and all the other problems formulated in their terms (let us call them all *classical problems* for the sake of simplicity) have no solution beyond propositional logic, given the undecidability of first-order logic (abbr.: FOL), a problem motivated by semantical structures known as *models* that, differently from proofs, which are finite by definition, may be infinite. Indeed, to say that *VAL* and *SAT* are formulated in semantical terms means that they are formulated in terms of *preservation of truth*: If all the, say, facts in a database are true, is a certain conclusion one wishes to draw therefrom always, or at least in some cases, also so? Given classical problems of very low complexity formulated in propositional logic, the semantical construct known as a truth table can provide a solution. But classical problems are more often than not highly complex, sometimes industrial-scale so, and they typically require a first- (or higher-) order language.

Fortunately, we have today a plethora of adequate proof systems for *VAL* and *SAT*. The Hilbert(-style) systems and the Gentzen systems, the latter divided into natural deduction and the sequent calculus, are proof systems to address *VAL*, and resolution and analytic tableaux are the two proof systems of election to find answers to classical problems formulated in terms of *SAT*. The comprehensive elaboration on these systems accounts for the expression "proofs" in our title, now complete as *Formal logic: Classical problems and proofs.* Although the first systems above are not algorithmic in nature, thus not providing efficient methods for classical problems, they are both historically and pedagogically relevant, and we accordingly discuss them in due detail. Resolution and analytic tableaux are at the root of many efficient SAT solvers, and we give equally full treatments of these calculi.

But there are more than these proofs. In the paragraph above we wrote "adequate" without brackets (compare with farther above), it being meant by this with respect to a proof system that one can prove in it every logical truth of the associated logic and nothing that is not a logical truth thereof. But these properties, known as completeness and soundness, require *metalogical proofs*—i.e. proofs at a level higher than the *logical proofs.* The same is true of the general undecidability of FOL, a result that is a celebrated answer to *VAL*. In turn, *VAL* and *SAT* have been proven to belong to specific classes of computational complexity—i.e. it has been shown how much they "cost"—, with these proofs constituting fundamental knowledge for the computational implementations of classical problems. Fulfilling our requirements of self-containment and comprehensiveness, we provide discussions of these celebrated proofs, as well as of the above-mentioned properties for all the proof systems we elaborate on in detail.

It is the moment now to convince the reader that ours is a truly original introduction to logic. Largely depending on the applications in view, logic can be approached today from three perspectives, to wit, mathematical, computational, or philosophical. Introductory textbooks to logic accordingly segregate their contents: Mathematical approaches typically concentrate on the mathematical properties of logical systems; computational approaches focus on computational implementations and automation of proofs; philosophical treatments greatly concentrate in argumentation. Gödel's (in)completeness and satellite results feature prominently in the first, as mathematical proof is a major concern of mathematical logic and it is unpalatable not to be able to prove a mathematical truth once one is discovered (or constructed, depending on one's philosophy of mathematics). The temporal and spatial costs of computational implementations, from the simple transformation of a formula into one acceptable by some software to the carrying out of a proof in it, are central topics in the second kind. Arguments, categorical syllogisms and fallacies included, occupy many of the pages of the third type. More technically, this can be reformulated as follows by invoking the four so-called *pillars of formal logic*: *Model theory* and *set theory* are major topics to be found in mathematical treatments of logic; *recursion*, or *computability*, *theory* features significantly in computational approaches; *proof theory* tends to be weighty in introductions to logic written for philosophy students. In particular, while the classical problems—*VAL* significantly less so than *SAT*—feature in introductory logic textbooks aimed at computer science students, they are largely or wholly absent from textbooks targeting a mathematical or philosophical studentship.

This segregation has constituted a successful recipe for a long time now, and possibly rightly so, but it does not reflect the current state of what can very generally be called formal logic. This book corrects this misguided state of affairs. Not focusing on the history of classical logic, this book nevertheless provides discussions and quotes central passages on its origins and development, namely from a philosophical perspective. Not being a book in mathematical logic, it takes formal logic from an essentially mathematical perspective. Biased towards a computational approach, with *SAT* and *VAL* as its backbone, this is thus an introduction to logic that covers essential aspects of the three branches of logic, to wit, philosophical, mathematical, and computational. More so, it gives practical applications of all these fields, namely in argumentation, theorem proving, logic programming, and even in logic design.

To be sure, the aim of reaching a large academic readership poses the risk of serving only a small one: The "traditional" tripartite segregation may in fact mirror some real distinctions, whether in skills or interests, in the different studentships. Moreover, the ambition of treating classical logic both at the object-language and at the metalanguage/metalogic levels while trying to keep the book in a "manageable" size may entail the suppression or obliteration of important contents of either of these components. To this we reply that no book stands alone, or is wholly self-contained; just as in any other field, certain treatments of logic have reached the status of standard works, and we refer to Hurley (2012), Mendelson (2015), and Boolos, Burgess, & Jeffrey (2007), for "classics" in philosophical, mathematical, and computational logic, respectively. Additionally, we hope the intersection of the above mentioned readerships is not empty. Our hope may in fact be a justified belief, as, for instance, linguists and computer scientists, to mention but these, may prove.

Be it as it may, we assume knowledge of, or at least familiarity with, mathematical concepts such as sets, functions, operations, and relations, providing solely definitions of less basic notions (e.g., Boolean algebra). In order to refresh their memory, or newly acquire such notions, mathematically literate readers can benefit from Bloch (2011) and the more mathematically reticent can do so from Makinson (2008). We also think that logic is a subject that requires both hands-on practice and reflection (or rumination), and we accordingly provide a vast selection of exercises ranging from the typical logic "drilling" exercise to commentary of relevant passages.

Finally: This book is in a large measure a selection, a restructuring, and an extension of contents first published in Augusto (2018). Main motivations for the present resulting text were the desire to improve, by reviewing and extending, the contents of the mentioned book, as well as the aim to provide a comprehensive stand-alone book on formal classical logic with the above-mentioned characteristics, in the belief that classical logic, particularly so in its formal version, is a subject both fascinating and–more and more–fundamental.

I wish to thank Dov M. Gabbay for accepting to publish this "extended remix," as well as Jane Spurr for her impeccable assistance as managing director of College Publications.

Madrid, Summer 2019

Luis M. S. Augusto