Dietrich, E., A. B. Markman, M. Winkley (2003). The Prepared Mind: The Role of Representational Change in Chance Discovery. In Yukio Ohsawa and Peter McBurney (eds.) *Chance Discovery by Machines*, Berlin: Springer-Verlag, pp. 208-230.

The Prepared Mind:

The Role of Representational Change in Chance Discovery

Eric Dietrich
Philosophy Dept.
Binghamton Univ.
Binghamton, NY 13902-6000
dietrich@binghamton.edu

Arthur B. Markman
Psychology Dept.
Univ. of Texas, Austin
Austin, TX 78712
markman@psy.utexas.edu

Michael Winkley
Philosophy Dept.
Binghamton Univ.
Binghamton, NY
mwinkley@binghamton.edu

"*Chance favors only the prepared mind*." - Louis Pasteur

Abstract

Analogical reminding in humans and machines is a great source for chance discoveries because analogical reminding can produce representational change and thereby produce insights. Here, we present a new kind of representational change associated with analogical reminding called *packing*. We derived the algorithm in part from human data we have on packing. Here, we explain packing and its role in analogy making, and then present a computer model of packing in a micro-domain. We conclude that packing is likely used in human chance discoveries, and is needed if our machines are to make their own chance discoveries.

1. Introduction.

A classic example of a chance discovery happened to the mathematician Henri Poincaré. In the late 1800s, Poincaré was working on what he termed Fuchsian functions (what we now call automorphic functions). One day, taking a break from his work, he took a small trip, and as he was about to board a bus, a new thought occurred to him. He relates "...just as I put my foot on the step, the idea came to me, though nothing in my former thoughts seemed to have prepared me for it, that the transformations I had used to define Fuchsian functions were identical to those of non-Euclidean geometry" (Poincaré, 1908). This insight turned out to be one of his more important mathematical discoveries.

Poincaré concluded that he had made this propitious discovery because his ideas and concepts had changed and altered in his head. Indeed, in his published works and speeches, he related many similar episodes that happened to him during his mathematical life, and in all of them he describes his ideas as jostling each other, coalescing, and in the process, changing.

What happened to Poincaré on that fateful day is what modern cognitive scientists call *analogical reminding*. His work on Fuchsian functions reminded him of transformations in non-Euclidean geometry. However, these two domains—Fuchsian functions and transformations in non-Euclidean geometry—are not similar in detail, but instead are *analogous*. We believe that many chance discoveries involve analogical remindings. So, in order to get our computers to make such discoveries, we need to implement in them a theory of analogical reminding. There is a robust and well-supported suite of theories, collectively called *Structure Mapping Theory*, about analogy and analogical reminding in humans which postulates that analogies are high-level, structural isomorphisms between knowledge representations (Gentner, 1983, 1989;

Gentner and Markman, 1997). Many of the theories in this suite have been implemented, and the results of the implementations compare well with human data (Falkenhainer, 1989; Forbus, et al., 1995). We will assume this suite of theories, which we explain below, because of the wealth of data supporting it, and because it is the dominant theory in the field of analogy research.

Structure mapping theory postulates that representational change occurs after the analogy is made (Gentner, 1983; Gentner and Wolff, 2000). But in cases of chance discovery, there are reasons to believe that at least some representational change has to occur before the analogy can be made (Dietrich, 2000). This sort of change is not currently included within structure mapping theory. In this paper, we extend the theory by incorporating within it a proposed mechanism for how knowledge representations change before an analogical reminding is completed. On our proposal, the chance reminding occurs *because* the representational change has been made. We call this mechanism for representational change *packing*.

2. Analogical Reminding and Structure Mapping Theory: Background.

Analogical reminding is a common psychological phenomenon in humans. It occurs any time some concept (or representation, we use the two terms interchangeably) in one domain facilitates recall of an analogous concept in another domain. A famous example of analogical reminding occurred when Ernst Rutherford was reminded of comets and their highly elliptical orbits around the Sun by studying the strongly deflected trajectories of alpha particles around nuclei. This analogy eventually led to the "solar-system" model of the atom. Another example is Kepler's analogy between light and the notion of the *anima motrix* (moving spirit). This analogy eventually led to Kepler's more refined analogy between light and the *vis motrix* (motive force), which postulated the Sun as the cause of the planet's orbits around it. The concept of the *vis motrix* was a precursor to the concept of gravity. Kepler knew that more distant planets moved more slowly in their orbits than could be predicted simply from their greater distances, and he had noticed that the planets moved more quickly

when they were closer to the Sun and more slowly when they were distant from the Sun -- a fact that could only be discerned by viewing the solar system as a heliocentric system. What would cause such a change in velocity? Kepler reasoned that this behavior could be nicely explained if the *anima motrix* emanated from the Sun and behaved like light: weakening with distance (see Gentner, et al., 1997 which examines this case in detail).

Before going further, we should state that we adopt the standard working assumption in cognitive science and AI about the composition of concepts. On this view, concepts are built from three main constituents: entities, attributes, and structural relations. Entities represent abstract or concrete objects. Attributes describe specific properties of entities. Relations provide information about the relationships among entities, attributes, or other relations. In mental representations constructed from these elements, attributes are denoted by unary predicates whose argument denotes the item described by that attribute. For example, *bright (x)* denotes that the argument to this attribute is bright. Relations are denoted by two or more place predicates having two or more arguments that specify the items that play particular roles within that relation. For example, in the relation *produce (x, y)*, the first argument is the producer and the second is the product.

Quite a bit is known about analogical reminding and the more inclusive class of similarity-based retrievals to which analogical reminding belongs. Analogical reminding is thought to comprise two processes: *access* and *mapping* (Forbus, et al., 1995, Hummel and Holyoak, 1997). Access (also simply called "retrieval") is the process of retrieving some representation from long-term memory based on some active representation that serves as a retrieval cue. In the usual case of analogical retrieval, the retrieving representation is the domain that we want to understand (or understand better). We refer to this domain as the *target*. The retrieved concept illuminates the target. We refer to it as the *base*. In Rutherford's analogy, the strongly deflected orbits of the alpha particles were the target domain, and the highly elliptical orbits of comets were the base domain.

Before the analogical retrieval, the target domain is accessible, but the base domain is not. After retrieval, both the base and target are accessible. The structure of the analogy between them may or may not be clear at this point, but it will be clear that the domains are similar in some way.

Mapping is the process of matching and placing in correspondence the constituents of the two active representations. Hence, mapping is a process of finding *functional counterparts* between concepts. Such functional counterparts are said to be in *structural alignment*. What makes these objects functional counterparts is that they play the same roles within a matching relational structure. It is this identity of role that is the basis of structural alignment.[1]

Mappings are found by determining the best match between representations that satisfies the constraints of *parallel connectivity*, *one-to-one mapping*, and *systematicity*. The parallel connectivity constraint ensures that when two elements are placed in correspondence, the arguments of those elements are also placed in correspondence. The one-to-one mapping principle states that each element in a domain can be mapped to at most one element in the corresponding domain. The systematicity constraint requires that, all else being equal, correspondences between *systems* of elements in the domains are preferred to matches between isolated elements. The mapping process is local-to-global. At the beginning of the comparison process individual entities, attributes, and relations are matched, and then the constraints are utilized to determine more global systems of relations. In sum, in making an analogy, the mapping process locates entities in the target domain that play the same roles (by being in identical structural relations) as the entities in the base domain.

Finally, we define two other technical terms of crucial importance. When a similarity or analogy comparison is made, elements (e.g., entities, attributes, and relations) of one concept are placed in correspondence with elements of another

[1] The mapping process in analogy tends to concentrate and preserve relational information over attribute information. But this happens only in analogy. In cases of ordinary similarity comparisons, attributes are also important (Gentner and Markman, 1997; Markman and Gentner, 1993).

concept. Some of the elements of the concepts are identical (of course, as we mentioned above, in an analogy, certain crucial structural relations have to be identical), but some are not. The elements that are not identical come in two major types. *Alignable differences* are nonidentical elements in corresponding positions in a system of relations. They are seen as similar -- aligned -- because of the similarity of the relational structure they participate in, rather than some inherent similarity of the items themselves. *Nonalignable differences* are nonidentical elements that are *not* in corresponding positions in any alignable system of relations and are therefore not seen as similar (Markman and Gentner, 1993, 1996). In figure 1, the pig and baby are alignable differences, as are the farmer and mother. The helicopter, barn, and hay in Figure 1a (top) are nonalignable differences relative the the bottom scene.

3. Why Representational Change is Needed to Understand Chance Discovery

Representational change is crucial to chance discovery, and analogical retrieval is responsible for particularly interesting kinds of representational change -- kinds that look well-suited to the task of chance discovery. Structure mapping theory postulates five kinds of change.

1. *Projection of candidate inferences*, in which structure from the base is transferred to the target, with appropriate substitutions when elements in the base and target correspond (Gentner, 1983, 1989; Markman, 1997).
2. *Progressive Alignment*, whereby children's knowledge becomes more abstract so that more high-order similarities can be recognized (Kotovsky and Gentner, 1996);
3. *Highlighting*, whereby less salient conceptual properties are made more salient;
4. *Rerepresentation*, whereby representations or either or both domains are changed to improve the analogical match;

5. *Restructuring*, whereby whole systems of knowledge get changed (Gentner and Wolff, 2000).

These kinds of representational change occur as a part of analogical mapping, and hence they happen after the base and target are both active. In the context of chance discovery, however, we are particularly interested in the kinds of representational change that occur in the process of retrieving information.

Previous implemented models of analogical retrieval have not assumed that any representational change occurs during the retrieval process (e.g., Forbus, Gentner, & Law, 1995; Thagard, Holyoak, Nelson, & Gochfeld, 1990). We believe that representational change must be an integral part of the analogical retrieval process based on what we call the *low-probability argument* (Dietrich, 2000). It seems unlikely that disparate domains that have not been compared previously already are represented using the same structure and yet that parallel structure had not been noticed. Clearly, there must be some similarity between domains to get a comparison started, but it is implausible that two domains are already represented with highly parallel structures before the target domain retrieves the base. Hence, retrieval models that do not posit representational change do not seem sufficiently powerful to explain the number of analogical remindings that occur both in extraordinary minds like Kepler's as well as in more ordinary ones.

If the high-level relational structures of the two analogues do not match before the analogy (except in very rare cases), but they do match at the time the analogy is made (and of course, afterwards), then some representational change must occur at the time of retrieval that permits an analogical alignment to occur. On this view, analogical retrieval is a constructive process, and not like taking a book of a shelf. The base and target representations may be altered in the process of retrieving an analog. Thus, analogical "retrieval" is not retrieval in the sense of getting something that is already there.

To illustrate the kind of representational changes the low-probability argument suggests, we consider two examples. The first is derived from experiments on analogical learning (Gentner, 1989). The second is about Kepler and his theory of planetary motion.

Assume that the idea that heat flows is new to a subject, S, but that S is well acquainted with flowing water. Suppose S is presented with a small, solid, silver bar sticking into a cup of hot coffee. And suppose that there is an ice cube on the end of the bar sticking out of the coffee. What happens to the ice cube? It rapidly melts. Suppose in contemplating this phenomenon, the following analogy occurs to S. S imagines a large beaker connected to a smaller beaker by some rubber tubing running from the base of the large beaker to the base of the small one. When the large beaker is filled with water, the water will flow from the large beaker into the smaller beaker through the tubing. The analogy is that heat flows from the hot coffee up the bar to the ice just as water flows from the large beaker through the tubing into the small beaker. The target domain is heat-flow, and the base domain is water-flow. The large beaker is analogous to the hot coffee, the small beaker is analogous to the ice cube, the tubing is analogous to the silver bar, and the flowing water is analogous to the moving heat. The core of the analogy is that the greater temperature of the coffee relative to the ice cube causes heat to flow up the bar just like the greater pressure of the water in the large beaker relative to the pressure in the small beaker causes the water to flow through the tube.

Gentner (1989), represented the base domain of water flow like this:

(CAUSE (GREATER
(PRESSURE large-beaker) (PRESSURE small-beaker))
(FLOW water tubing large-beaker small-beaker))

A parallel relational structure was used to represent heat flow:

(CAUSE (GREATER
(TEMPERATURE coffee) (TEMPERATURE ice-cube))
(FLOW heat bar coffee ice-cube))

The two uses of the relation FLOW(w, x, y, z) have to be identical in order for the analogy to be constructed. But according to the low-probability argument, the above two representations must be viewed as *end*-products in S's representations of the domains. Before the analogy occurred to S, it is unlikely that the flow of heat was conceptualized by S in the same way as the flow of water. Otherwise, there would have been nothing to discover. Instead, flowing water was probably conceptualized in a manner tied closely to knowledge of liquids. Therefore, in order for the analogy between heat flow and water flow to occur, the concept of FLOW had to become more abstract, losing its relationship to specific properties of liquids. There are several ways that a concept might change to allow such abstraction. We consider two of them here.

The first type of conceptual change involves broadening the range of items to which a particular predicate could apply. For this example, assume that our intrepid scientist, S, did conceive of heat as something that moves. This conception was likely different from S's conceptualization of water. There are probably several different kinds of movement predicates, such as one for describing the movement of stuff like water, and one for other kinds of more abstract movements like the flow of emotion through a person or a group of people or the flow of information through a community. In this case, let us assume, there is a predicate *L-flow* (L for "liquid") that would have one argument for the substance that is flowing. The value of this argument would be restricted to physical matter in a liquid state, specifically to a unified body of matter of no particular shape with a tendency to flow. This restriction could be implemented in many different ways, the easiest being a data type restriction on the type of argument the predicate *L-flow* can take. Thus, *L-flow* involves a particular attribute of one of its arguments, and it is these attributes that change.

Because of this restriction, *L-flow* cannot apply to heat or to emotion (or to other movements that seem less like flow such as a person walking down the street or a ball thrown through the air). In order to recognize that the movement of heat or emotion is like the movement of liquid, it is necessary to generalize the concept by removing the type restriction on *L-flow*. Even less radical types of flow, such as the flow of granular masses like flour, sugar, or dirt, would require some loosening of the type restriction.

It is possible that comparing types of movement might suggest that a particular type restriction could be relaxed. For example, the physical similarity of water pouring from a glass and sugar pouring from a bowl might be sufficient to recognize that the substance that flows need not be a liquid. In this case, the type restriction is altered. Then, when flow is applied to more abstract concepts like heat or emotion, two things are likely to occur. First, the type restriction on the argument is likely to become even more loose and abstract. Second, the representation of the more abstract concepts may also change to include some attributes that make them more similar to other items that flow. Indeed, early theories of heat conceptualized it as a liquid or a particulate substance (e.g., Wiser & Carey, 1983). This type of representational change involving attribute change will be considered further in section 5; it is, we claim, the basic kind of packing.

The second case of conceptual change during analogy involves the creation of a new predicate as a result of the juxtaposition of mismatching items. In this case, assume that S has not yet conceptualized heat as something that can flow. Hence, the concept of flow would only be applied to describe the movements of physical substances. The movement of heat would require some other predicate. For example, S might represent heat using a general predicate like "moves," which would yield a representation like the following.

(CAUSE (GREATER
(TEMPERATURE coffee) (TEMPERATURE ice-cube))
(MOVE heat bar coffee ice-cube))

The analogy could still go through, however, because there is enough similarity between the above representation and the representation for flowing water. What is crucial about this case is the predicate change from "moves" to "flows." "Move" is too general. Planets, feet, and money move, but they don't flow. Still, in order for the predicate "flow" to replace the predicate "move" in this context, "flow" must be abstracted; it must lose information. This is because it is used for standard liquids. Changing from "move" to "flow" allows S to understand heat as some sort of unified stuff (but unlike matter) that slowly climbs its way up the bar from the hot coffee. Changing from "move" to "flow" is a particularly robust representational change. This case will also be considered further in section 5; it requires a special kind of packing.

We now return to Kepler. Kepler was an Aristotelian until his dying day, meaning that he had no notion of inertia and believed that things move only because they are pushed; if something is not pushed, it ceases to move. In his first book, the *Mysterium Cosmographicum* (1596), Kepler asked this question concerning the planetary orbits: why did the outer planets move more slowly than the inner planets, even factoring in there great distance? Saturn's orbit is about twice the size of Jupiter's, but Saturn takes substantially more than twice as long to go around once -- about two and half times as long. His answer was:

> "...we must choose between two assumptions: either the souls which move the planets are less active the farther the planet is removed from the sun, or there is only one moving soul in the center of all the orbits, that is the sun, which drives the planet the more vigorously the closer the planet is, but whose force is quasi-exhausted when acting on the outer planets because of the long distance and the weakening of the force which it entails."

He chose the second and thereby was analogically reminded of light (it is not known that the analogical reminding occurred right then; in fact it is not known when the analogy occurred to him, but it is reported in the *Mysterium*). In short order, because of

his insight, the reason for the planets' movements went from being souls (*anima motrix*) to being a mechanical, physical cause (*vis motrix*). This reconceptualization of planetary motion set the stage for Newton's proposal for the role of a gravitational force in planetary motion.

In making this analogy, however, Kepler had to suppress a lot of what he knew about light. The *vis motrix* had to somehow whip the planets around, veering at a large angle from the direction of emanation from the Sun. But light does not work this way at all. It leaves the Sun and reaches all the planets in a straight line beginning at the Sun and ending at the side of the planet facing the Sun. Indeed this is more of a problem for Kepler's conception of the *vis motrix* than it is for the *anima motrix* because for the latter, one can assume a modicum of intelligence: the moving soul *knows* it has to push the planets, the moving force doesn't. Furthermore, light from the Sun illuminates everything from planets to the noses on our faces. But the *vis motrix* apparently only operates on planets (and not noses). Again, information crucial to understanding light from the Sun had to be suppressed in order for the analogical mapping to occur; that is, information had to be suppressed or packed.

4. Packing: An Overview

Packing is a process of representational change during memory retrieval that suppresses irrelevant information in a structured mental representation. This suppression alters the structure of the representation by making aspects that are not relevant to the correspondence between base and target less accessible. Though we explore packing as it relates to analogical reminding, packing is not restricted to analogical reminding. Rather, it is a general memory process that simplifies structured representations.

We have done some experiments on memory for pairs of complex scenes following comparisons of those scenes, and the data suggest that attributes of objects are packed when those attributes are irrelevant to the relational match among items

(Stilwell, Markman, & Dietrich, 2001, 2002). Though we only have data indicating that irrelevant attribute information is packed, the computer model described below extends packing to handle packing of whole relational predicates, as well as predicate *change*. These types of representational change parallel the two processes described above about heat flow.

Packing is a syntactic process, and is domain-general. The packing process does not care about the semantic content of the representation (such as the labels or inferential connections or the information content, etc.). This contrasts with many representational processes, which are domain-specific and require lots of domain knowledge to be carried out.

The packed representations are more *abstract* than their less abstract versions. By "abstract," we mean that the representations with packed components contain less information than their unaltered versions. This usually means that the representations have wider applicability than their unpacked cousins.

Finally, packed information can be *unpacked*. That is, the suppressed information can be recovered, made explicit, and used for other reasoning tasks such as inferencing.

5. STRANG: A computational model of packing during analogy making.

Our model of packing in analogical reminding is called STRANG, short for "STRing AnaloGizer." It is implemented in Java, and operates over letter strings (e.g., abbccc). STRANG has only limited knowledge of the English alphabet. In particular, it knows the shapes of the letters (in one font only) and their sequence. The shapes shown in Figure 2. The labels on the segments of the template correspond to attributes that are used to describe the letters. STRANG's knowledge of the sequence of the letters is limited to the ability to produce the successor of a letter presented.

There are two reasons why letter strings are a good arena for an initial exploration of packing. First, by using such a sparse domain, the representations need

not be judged on the basis of their match to what people know about the domain. Thus, our representational assumptions can be made explicit. It also prevents us from reading too much into the content of the predicates. It is tempting to see meaningful predicate names in knowledge representations as actually representing what they name, i.e., to assume that a node labeled "dog" actually represents dogs. It is then also tempting to assume that a knowledge representation actually models the relevant concept, i.e., that a dog-node (or a dog-frame, or whatever) is an accurate model of a human's concept of a dog (Palmer, 1978). Both of these assumptions are unjustified and should be avoided. In fact, no one currently knows the actual content or structure of mental representations in the human mind, hence no one is sure how to mirror human representations in computers (Markman, 1999). Second, because these packing processes are complex, a simple domain provides us with a small number of predicates and hence keeps the packing process managable so we are able to evaluate the success of the model. If there were too many predicates in the domains, it might be difficult to determine why the model acts as it does.

Strictly speaking, STRANG makes analogies during a process of analogical retrieval: given a string in working memory, STRANG retrieves an analogous string from long-term memory. But STRANG is not intended as a model of analogical retrieval. The essential elements of STRANG are those that pack the representations. STRANG has to make retrievals in order to exhibit representational change via packing. The essential parts of STRANG could be included in other, more psychologically accurate, models of analogical retrieval such as Forbus et al.'s, 1995, MAC/FAC or Thagard et al.'s, 1990, ACME.

5.1 Background on the operation of STRANG

STRANG has two sets of letter strings. First, there is a collection of letters in long-term memory. There are no connections among strings in long-term memory. Second, there is a current active string that serves as a retrieval cue. Both the strings in long-term memory and the one serving as a cue are represented by structured representations consisting of attributes that describe the letters in the string as well as

relations that describe the construal of the string. Before we can describe the operation of STRANG, we must first describe the representational system in more detail.

STRANG represents letters as entities with collections of attributes. Most of these attributes are derived from viewing letters in a grid-like font where each line segment of the grid is given a designation. Figure 2 shows the set of segments that describe the letters as well as an image of the letters themselves. Here are the letters we use in the examples:

a - LET, L00, L01, L02, L11, L18

b - LET, L00, L02, L06, L09, L11

c - LET, L00, L02, L09

e - LET, L00, L01, L02, L09, L17

f - LET, L00, L04, L07, L10

g - LET, L00, L02, L03, L09, L11, L14

h - LET, L00, L06, L09, L11

i - LET, L10, L26

j - LET, L05, L10, L13, L26

m - LET, L00, L09, L10, L11

n - LET, L00, L09, L11

o - LET, L00, L02, L09, L11

p - LET, L00, L02, L09, L11, L12

q - LET, L00, L02, L09, L11, L14

r - LET, L00, L09

(where the attribute LET is common to all letters, and designates that the entity is a letter)

All strings (both the cues and those in long-term memory) are parsed by creating a parse tree of descriptions. When analyzing a string, the category name is used as the root node of the parse tree. Parse trees (not necessarily binary) will be represented as lists: (root, leftmost-subtree, next-subtree, . . . rightmost-subtree).

When parsed, there are five types of categories into which strings are placed:

(1) *sequence* (abbreviated: Seq)
(2) *identity group* (abbreviated: IG)
(3) *repeating sequence* (abbreviated: Rseq)
(4) *repeating group* (abbreviated: RG)
(5) *string* (abbreviated: ST)

[Note: this category is used when none of the above four correctly describe the vector of letters.]

The main relationship governing the formation of complex strings (i.e., strings with multi-letter substrings as parts) is the relation *Concats-with* (i.e., *letter-string1 concatenates with letter-string2*). This relation is a 2-ary relation.

As mentioned, categories have attributes (single-place predicates) modifying them. These attributes are:

(1) Constituent= <string category> (i.e., *The constituents of this string are of these categories*)
(2) Card <integer> (i.e., *This string has cardinality <integer>*)
(3) Begins-with <letter> (i.e., *This string begins with this letter*)

Not all attributes are always present, especially after packing.

All strings are made up of letters in binary relations. The three relations are:

(1) IAF <letter1, letter2>

(i.e., *letter1 immediately alphabetically follows letter2*)

[Note: this relation was chosen over "immediately proceeds" because in English we read from left to right, but it also forces the parse to list the strings backwards.]

(2) IW <letter1, letter2>

(i.e., *letter1 is identical with letter2*)

(3) OW <letter1 letter2>

(i.e., *letter1 occurs with letter2*)

[Note: this is the default relation letters enter into when the above two are not applicable.]

Finally, in the descriptions of the parses below:

(1) Entities are enclosed in quotes,

(2) Attributes will be enclosed in angle brackets and are to be read as attributes of the entities they come after,

and

(3) Relations are underlined.

5.2 Packing irrelevant attribute information

Suppose STRANG gets as input the repeating sequence *abab* and has in memory the repeating sequence *efef* (and suppose here, for simplicity, that only *efef* is in memory so that it will be retrieved). Here are the representations of these two strings.

abab =

("Rseq" <(card 2) (Constituent= seq, begins-with "a", card 2)>
(Concats-with
("Seq" <(card 2) (begins-with a)>
(IAF ("b" <LET>, <L00>, <L02>, <L06>, <L09>, <L11>)
("a" <LET>, <L00>, <L01>, <L02>, <L11>, <L18>)))
("Seq" <(card 2) (begins-with a)>
(IAF ("b" <LET>, <L00>, <L02>, <L06>, <L09>, <L11>)
("a" <LET>, <L00>, <L01>, <L02>, <L11>, <L18>)))))

efef =

("Rseq" <(card 2) (Constituent= seq, begins-with "e", card 2)>
(Concats-with
("Seq" <(card 2) (begins-with "e")>
(IAF ("f" <LET>, <L00>, <L04>, <L07>, <L10>)

("e" <LET>, <L00>, <L01>, <L02>, <L09>, <L17>)))
("Seq" <(card 2) (begins-with "e")>
(IAF ("f" <LET>, <L00>, <L04>, <L07>, <L10>)
("e" <LET>, <L00>, <L01>, <L02>, <L09>, <L17>)))))

Note how entities and relations alternate. Note also that all the entities at any level of abstraction have attributes. And finally note that the attributes of "seq" and "Rseq" are derived from the relational structures they dominate, together with the attributes of those structures. We call this "promotion of attributes."

There is plenty of relational structure between the above to representations to give us an analogy between them. Indeed, *abab* and *efef* are intuitively analogous (in string world, anyway; in the ordinary world, they might be regarded as too representationally thin to be true analogies). However, the irrelevant attribute information could get in the way of the analogy, for the representations are quite detailed. To the extent that the attributes in the two representations are taken seriously by STRANG, there really cannot be an analogy between them, for the representations don't in fact look at like. Of course, we can see that the representations *do* look alike, but that is just because we are looking past the details: we are *packing away* the irrelevant details. This is exactly what packing is for. For example, being a sequence beginning with "a" is not like being a sequence beginning with "e" unless what the sequence starts with doesn't matter.

Now for the central idea behind STRANG - to form the analogy between *abab* and *efef*, the irrelevant attribute information is packed away. We define *irrelevant attribute information* as the information that two objects do *not* have in common during a similarity comparison. The packed information is stored in *packed variables*, denoted by lower-case Greek letters. The packed variables contain the irrelevant attribute information. This now gives:

packing of *abab* relative to *efef* =

("Rseq" <(card 2) α>
  (Concats-with
    ("Seq" <(card 2) β>
      (IAF ("b" <LET>, <L00>, χ)
           ("a" <LET>, <L00>, <L01>, <L02>, δ)))
    ("Seq" <(card 2) β>
      (IAF ("b" <LET>, <L00>, χ)
           ("a" <LET>, <L00>, <L01>, <L02>, δ)))))

packing of *efef* relative to *abab* =

("Rseq" <(card 2) κ>
  (Concats-with
    ("Seq" <(card 2) ϕ >
      (IAF ("f" <LET>, <L00>, γ)
           ("e" <LET>, <L00>, <L01>, <L02>, η)))
    ("Seq" <(card 2) ϕ >
      (IAF ("f" <LET>, <L00>, γ)
           ("e" <LET>, <L00>, <L01>, <L02>, η)))))

The representations are now more parallel than they were before. The packed variables differ, of course, but they can now be easily excluded from the similarity match because of their special status as packed variables. Given this, the above two representations are now *analogous*. In this case, some of the attributes are retained. In particular, the shared attributes that make up the letters are kept. This seems plausible because when people make analogies, some shared attributes are kept. For example: from Figure 1, a mother yelling at her baby who is making a mess by drawing on the walls and floors and a farmer yelling at her mess-making pig have some attributes in common. Both the mother and the farmer are yelling ("yelling-at" is a relation, but "yelling" is an attribute), and both the baby and the pig are living things that can make a

mess and, in fact, are making a mess. So it seems that these, but only these, attributes are retained.

As another example, consider *abab* and *efgefg*; are they analogous? They are clearly similar (they are both repeating sequences), but do they share enough structure to be analogies?

abab =
("Rseq" <(card 2) (Constituent= seq, begins-with "a", card 2)>
(Concats-with
("Seq" <(card 2) (begins-with a)>
(IAF ("b" <LET>, <L00>, <L02>, <L06>, <L09>, <L11>)
("a" <LET>, <L00>, <L01>, <L02>, <L11>, <L18>)))
("Seq" <(card 2) (begins-with a)>
(IAF ("b" <LET>, <L00>, <L02>, <L06>, <L09>, <L11>)
("a" <LET>, <L00>, <L01>, <L02>, <L11>, <L18>)))))

efgefg =
("Rseq" <(card 2) (Constituent= "Seq", begins-with "e", card 3)>
(Concats-with
("Seq" <(card 3) (begins-with "e")>
(IAF ("g" <LET>, <L00>, <L02>, <L03>, <L09>, <L11>, <L14>)
("f" <LET>, <L00>, <L04>, <L07>, <L10>))
(IAF ("f" <LET>, <L00>, <L04>, <L07>, <L10>)
("e" <LET>, <L00>, <L01>, <L02>, <L09>, <L17>)
("Seq" <(card 3) (begins-with "e")>
(IAF ("g" <LET>, <L00>, <L02>, <L03>, <L09>, <L11>, <L14>)
("f" <LET>, <L00>, <L04>, <L07>, <L10>))
(IAF ("f" <LET>, <L00>, <L04>, <L07>, <L10>)
("e" <LET>, <L00>, <L01>, <L02>, <L09>, <L17>)

Now, upon packing, we get:

packing of *abab* relative to *efgefg* =
("Rseq" <(card 2) α>
(Concats-with
("Seq" <β>
χ)
("Seq" <β>

χ)

packing of *efgefg* relative *abab* =

("Rseq" <(card 2) δ >

(Concats-with

("Seq" <ϕ >

γ)

("Seq" <ϕ >

γ)

We see that the two remaining abstract structures are rather thin, indeed. Even the letters and letter structures have been packed because a sequence of three and a sequence of two are too dissimilar. Basically, there is one relationship governing this similarity comparison: namely that each string is composed of two sequences, i.e., each is a repeating sequence. In this case, the relational nodes that have IAF as their roots were also packed away on the grounds that they are nonalignable differences. Thus STRANG explicitly packs away more than irrelevant attribute information. Here is one place where STRANG goes beyond the experiment data. In the next section, we discuss another, more extreme case.

5.3 Packing and predicate change

Now let's turn our attention to predicate change. We have no human data on this kind of representational change, so here is where STRANG goes significantly beyond the experimental results.

We consider the analogy between *ababccc* and *mnopqrhijhijhij*. First, here are the relevant representations.

ababccc =
("ST" <(card 2) (Constituents= "Rseq"; "IG")>
(Concats-with
("Rseq" <(card 2) (Constituent= "Seq", begins-with "a", card 2)>

(Concats-with
("Seq" <(card 2) (begins-with "a")>
(IAF ("b" <LET> <L00> <L02> <L06> <L09> <L11>)
("a" <LET> <L00> <L01> <L02> <L11> <L18>)))
("Seq" <(card 2) (begins-with "a")>
(IAF ("b" <LET> <L00> <L02> <L06> <L09> <L11>
("a" <LET> <L00> <L01> <L02> <L11> <L18>)))))
("IG" <(card 3) (begins-with "c")>
(IW ("c" <LET> <L00> <L02> <L09>)
("c" <LET> <L00> <L02> <L09>)
(IW ("c" <LET> <L00> <L02> <L09>)
("c" <LET> <L00> <L02> <L09>)))))

mnopqrhijhijhij =
("ST" <(card 2) (Constituents= "Seq"; "Rseq")>
(Concats-with
("Seq" <(Card 6) (begins-with "m")>
(IAF ("r" <LET> <L00> <L09>)
("q" <LET> <L00> <L02> <L09> <L11> <L14>))
(IAF ("q" <LET> <L00> <L02> <L09> <L11> <L14>)
("p" <LET> <L00> <L02> <L09> <L11> <L12>))
(IAF ("p" <LET> <L00> <L02> <L09> <L11> <L12>)
("o" <LET> <L00> <L02> <L09> <L11>))
(IAF ("o" <LET> <L00> <L02> <L09> <L11>)
("n" <LET> <L00> <L09> <L11>))
(IAF ("n" <LET> <L00> <L09> <L11>)
("m" <LET> <L00> <L09> <L10> <L11>)))
("Rseq" <(card 3) (Constituents= "Seq", begins-with "h", card 3)>
(Concats-with
("Seq" <(card 3) (begins-with "h")>
(IAF ("j" <LET>, <L05, <L10, <L13, <L26>)
("i" <LET>, <L10>, <L26>))
(IAF ("i" <LET>, <L10>, <L26>)
("h" <LET>, <L00, <L06, <L09, <L11>))
("Seq" <(card 3) (begins-with "h")>
(IAF ("j" <LET>, <L05, <L10, <L13, <L26>
("i" <LET>, <L10, <L26>)
(IAF ("i" <LET>, <L10, <L26>)
("h" <LET>, <L00>, <L06>, <L09>, <L11>))
(Concats-with
("Seq" <(card 3) (begins-with "h")>
(IAF ("j" <LET>, <L05>, <L10>, <L13>, <L26>)
("i" <LET>, <L10>, <L26>))
(IAF ("i" <LET>, <L10>, <L26>)
("h" <LET>, <L00>, <L06>, <L09>, <L11>)))
("Seq" <(card 3) (begins-with "h")>
(IAF ("j" <LET>, <L05>, <L10>, <L13>, <L26>)
("i" <LET>, <L10>, <L26>))

(IAF ("i" <LET>, <L10>, <L26>)
("h" <LET>, <L00>, <L06>, <L09>, <L11>)))))))

Basic STRANG aligns (finds an analogy between) the two repeating sequences, *abab* and *hijhijhij* , from the two original strings (this assumes that *ababccc* is the base domain). Using STRANG's standard packing algorithm, the packing the above two representations relative to each other would produce:

packing of *ababccc* relative to *mnopqrhijhijhij* =
("ST" <(card 2) α>
(Concats-with
"Rseq" <β>
χ))

and

packing of *mnopqrhijhijhij* relative to *ababccc* =
("ST" <(card 2) δ >
(Concats-with
ϕ
"Rseq" <γ>))

These packings do not preserve much information. That is unfortunate, because there is additional similar structure if one looks more deeply. To get at this structure, STRANG uses an algorithm called the *cookie cutter* -- so called because the structure of one representation is used to *re*-structure the other like a cookie cutter cutting out cookies in dough. The cookie cutter is only called during certain situations: STRANG measures how much structure is in the packed representations it creates and if that amount is low enough, then in a small percentage of those cases, the cookie cutter is called.

When the cookie cutter is called in the case of *ababccc* and *mnopqrhijhijhij* , then the substrings *abab* and the *mnopqr* are *forced* to have similar structure by altering their

predicates. Then the *ccc* and the *hijhijhij* naturally align due to their having the same cardinality. Here is what the cookie cutter produces in this case:

packing of *ababccc* relative to *mnopqrhijhijhij* =
("ST" <(card 2) (Constituent= "?_NewNode" "β")>
(Concats-with
("?_NewNode" <(card 2)>)
(Concats-with
"η"
"φ"
("β" (card 3)))

packing of *mnopqrhijhijhij* relative to *ababccc* =
("ST" <(card 2) (Constituent= "?_NewNode" "δ")>
(Concats-with
("?_NewNode" <(card 2)>)
(Concats-with
"μ"
"λ"))
("δ" <(card 3)>)))

There are several things to note about these representations. First and foremost, there is the new node, *?_NewNode*. This node is a second kind of packed variable which is an unnamed new entity, i.e., it designates a new category that STRANG doesn't have detailed knowledge of, hence the "*?*". Second, note that this category has a cardinality of 2, so the relevant cardinality attribute changed from 6 to 2. This means *mnopqr* has been re-represented as (*mno*)(*pqr*). This is quite a radical restructuring of the representation, and is accomplished by changing the cardinality predicate and packing away the string *mnopqr* as (*mno*)(*pqr*). STRANG has no category for a list or group of non-repeating sequences. But because of the re-representation, STRANG has the beginnings of such a category after the analogy. If need be, STRANG also can generate representations of these two strings with more semantics than the above two

representations have, thus making the fact that the two constituents are sequences explicit. This is accomplished by attaching to the list of attributes of *?_NewNode* the predicate *(Constituents= "Seq")*, giving us:

("?_NewNode" <(card 2) (Constituents= "Seq">).

Finally, note that the remaining substrings, *ccc* and *hijhijhij*, are aligned because they each have cardinality 3. This means that STRANG has packed away all the other information and has just focused on the fact that these strings have the same number of constituents.

6. Chance Discoveries and the Prepared Mind.

Our argument for the importance of packing in chance discovery is simple. We showed in section 3 that packing is crucial to making at least some analogical remindings. And analogical reminding is crucial to at least some chance discoveries. Hence, packing is likely to be crucial to chance discovery.

It remains to be seen whether similar mechanisms can be used to understand machines. As discussed in the previous section, interesting and useful chance discoveries require seeing two separate domains of knowledge as interestingly analogous at a new abstract level, which is more abstract than the level at which the domains were learned. A mind prepared for chance discoveries, whether natural or artificial is a mind capable of packing and analogical reminding. Without these capacities, chance discoveries are unlikely to occur.

However, it seems that packing via the cookie cutter is quite powerful: virtually any two strings can be made analogous to each other using this approach. It is legitimate to worry, then, that packing may be *too* powerful. If packing could render any two concepts analogous, then chance discoveries would be virtually impossible, because sifting through the enormous quantities of fruitlessly analogous concepts would swamp

any intelligent system. However, the strength of packing is partly an artifact of the domain. Packing is very powerful in the string micro-world precisely because this domain is so simple and therefore produces extremely simple representations -- parse trees are nowhere near as complicated as real human concepts or the knowledge representations needed in real-world AI. It is unlikely that packing is as strong a mechanism in humans as it is in STRANG. That said, packing is still a powerful technique for producing analogies via reminding. Even in humans, it does seem as if it licenses analogies between a very large number of potential analogical correspondences. This could still be a serious problem for our theory, for the number of useful, chance discoveries is quite small, and it is not clear how any of them are ever discovered if a large portion of computing time is spent packing representations away in a myriad of different combinations.

To solve this problem, we must delve more deeply into the mechanisms of analogical reminding, and then explore the factors that make analogies useful. Data on human reminding show that the majority of remindings are based on surface similarities (e.g., similarities in the letters in the strings rather than similarities in the relations). Yet analogies based on structural similarities do occur from time to time. These two facts are the basis of one of the most prominent extant models of analogical retrieval, the MAC/FAC model. MAC/FAC stands for "Many are called but few are chosen." The first stage of this model involves a cheap similarity computation between semantic vectors composed of the predicates that appear in the representations of a variety of domains. This vector similarity calculation can be carried out efficiently, even for large knowledge-bases. This first stage is used to weed out all but a few possible analogical remindings. Then, the second stage performs the mapping stage of analogy finding using the Structure Mapping Engine. This model tends to produce remindings that share substantial surface similarity with the cue.

We are exploring two different alternatives. The first is that packing is assumed to work independently of this similarity-based reminding process. In the context of chance discovery, packing has to be more opportunistic. Because the process is computationally expensive and produces a variety of potential matches given a pair of domains, it is unlikely to be fruitfully applied to a whole knowledge-base. However, in

the context of scientific discovery, there are many opportunities for packing to operate on domains that happen to be available for other reasons. For example, Kepler was exposed to William Gilbert's nascent concept of magnetism, and spent considerable time working out the analogy between magnetism and the *vis motrix*. These opportunistic juxtapositions provide cases for the packing mechanism to operate on.

The second is that perhaps STRANG could be made part of MAC/FAC. Because MAC/FAC starts with the computationally cheap calculation and then moves on to the more complex calculation, STRANG could be inserted between them, creating a retrieval program that would do a reduced form of structure mapping with abstraction via packing during retrieval. STRANG would, in effect, do a second, packing based "retrieval" -- a kind of filtering -- on the representations retrieved first by MAC, and then output these filtered representations to FAC.

A final important point to address is whether the analogies that emerge from STRANG will be useful. This issue is important, because the representations that emerge from the packing process are guaranteed to be analogous; they are constructed to create parallel relational structures between domains. Whether these matches are important for science (or anything else) depends on several other considerations, some of which have to do with the knowledge possessed by the human (scientist) having the analogy, and some having to do with the nature of the world. Whether an analogy turns out to be useful depends on the candidate inferences and other kinds of knowledge change the scientist can muster. This is purely a matter of what knowledge the system possesses. Kepler was obviously very good at developing analogies and mining them for insights. However, many profound analogies in science don't pan out not because the scientist lacks some requisite knowledge but because the world doesn't conform to the predictions of the analogy. These analogies are seldom reported, but again, Kepler's research provides a good example: his analogy between the *vis motrix* and magnetism was brilliant but incorrect. As it turns out, gravity is not like magnetism. All of this means that *chance* really does play a large role in chance discoveries, which in turn means that the kinds of analogies that result in profound scientific chance discoveries on our model will be rare -- which they are.

A very interesting research project would be to add packing and the rest of the associated analogical retrieval algorithms to knowledge discovery and data mining machines. There is much research to do on optimizing the packing algorithm. Packing could be placed in different parts of the retrieval process. And, finally, packing could be made a background process going on constantly between knowledge representations in long-term store. All of these options remain to be explored.

7. Conclusion.

We have proposed extending the structure-mapping theory of analogy and analogical retrieval by adding a process of packing parts of representations away, thereby producing more abstract concepts. We have detailed a model, based on human data, that implements packing for analogical retrieval. We have shown that packing can introduce new representational structure with new semantics. And, finally, we have shown where packing fits into the broader theoretical foundation for analogical retrieval.

Really understanding chance discovery will require understanding how representations change. Unfortunately, it has proven difficult to make headway on mechanisms for representational change. We believe that the packing mechanisms described here will help change this situation and greatly enhance reasoning systems that are devoted to making new discoveries. These mechanisms will permit automated systems to make use of the power of analogy to create new knowledge.

Dietrich, E. (2000). “Analogy and conceptual change, or you can't step into the same mind twice." In E. Dietrich and A. Markman (eds.), *Cognitive Dynamics: Conceptual and representational change in humans and machines*. Mahwah, NJ: Lawrence Erlbaum, pp. 265-294.

Falkenhainer, B., Forbus, K., & Gentner, D. (1989). "The structure-mapping engine: Algorithm and examples," *Artificial Intelligence 41 (1)*, pp. 1-63.

Forbus, K., Gentner, D., & Law, K. (1995). "MAC/FAC: A model of similarity-based retrieval." *Cognitive Science* 19, pp. 141-205.

Gentner, D., Brem, S., Ferguson, R., Markman, A., Levidow, B., Wolff, P., & Forbus, K. (1997). "Analogical reasoning and conceptual change: A case study of Johannes Kepler." *Journal of the Learning Sciences*, 6(1), pp. 3-40.

Gentner, D. (1983). "Structure-mapping: A theoretical framework for analogy," *Cognitive Science 7*, pp. 155-170.

Gentner, D. (1989). "The mechanisms of analogical learning," In S. Vosniadou & A. Ortony (eds.) *Similarity and analogical reasoning*. Cambridge, UK: Cambridge Univ. Press, pp. 199-241.

Gentner, D. & Markman, A. B. (1997). "Structure mapping in analogy and similarity." *American Psychologist*, 52(1), pp. 45-56.

Gentner, D. & Wolff, P. (2000). "Metaphor and knowledge change,” In E. Dietrich and A. Markman (eds.), *Cognitive Dynamics: Conceptual and representational change in humans and machines*. pp. 295-342.

Hummel, J. E. & Holyoak, K. J. (1997). "Distributed representations of structure: A theory of analogical access and mapping." *Psychological Review*, 104(3), pp. 427-466.

Kotovsky, L. & Gentner, D. (1996). "Comparison and categorization in the development of relational similarity." Child Development 67, pp. 2797-2822.

Markman, A. B. (1997). "Constraints on analogical inference." *Cognitive Science*, 21(4).

Markman, A. B. (1999). *Knowledge Representation*. Lawrence Erlbaum.

Markman, A. B. & Gentner, D. (1993). "Structural alignment during similarity Comparisons." *Memory and Cognition*, 24(2), pp. 235-249.

Markman, A. B. & Gentner, D. (1996). "Commonalities and differences," *Memory and Cognition*, 24, pp. 235-249.

Markman, A. B. & Gentner, D. (1997). "The effects of alignability on memory." Psychological Science, 8(5), pp. 363-367.

Palmer, S.E. (1978). Fundamental aspects of cognitive representation. In E. Rosch & B.B. Lloyd (Eds.) *Cognition and Categorization* (pp. 259-302). Hillsdale, NJ: Lawrence Erlbaum Associates.

Poincare, H. (1908/1952). *Science and Method*, Dover Publications, New York.

Stilwell, C. H., Markman, A. B., Dietrich, E. (2001). "The Fate of Irrelevant Information in Analogy", Unpublished Ms.

Thagard, P., Holyoak, K.J., Nelson, G., & Gochfeld, D. (1990). Analogical retrieval by constraint satisfaction. *Artificial Intelligence, 46*, 259-310.

Tulving, E. & Thompson, D. M. (1973). "Encoding specificity and retrieval processes in episodic memory." *Psychological Review*, 80, pp. 352-372.

Wiser, M., & Carey, S. (1983). When heat and temperature were one. In D. Gentner & A.L. Stevens (Eds.) *Mental Models* (pp. 267-298). Hillsdale, NJ: Lawrence Erlbaum Associates.

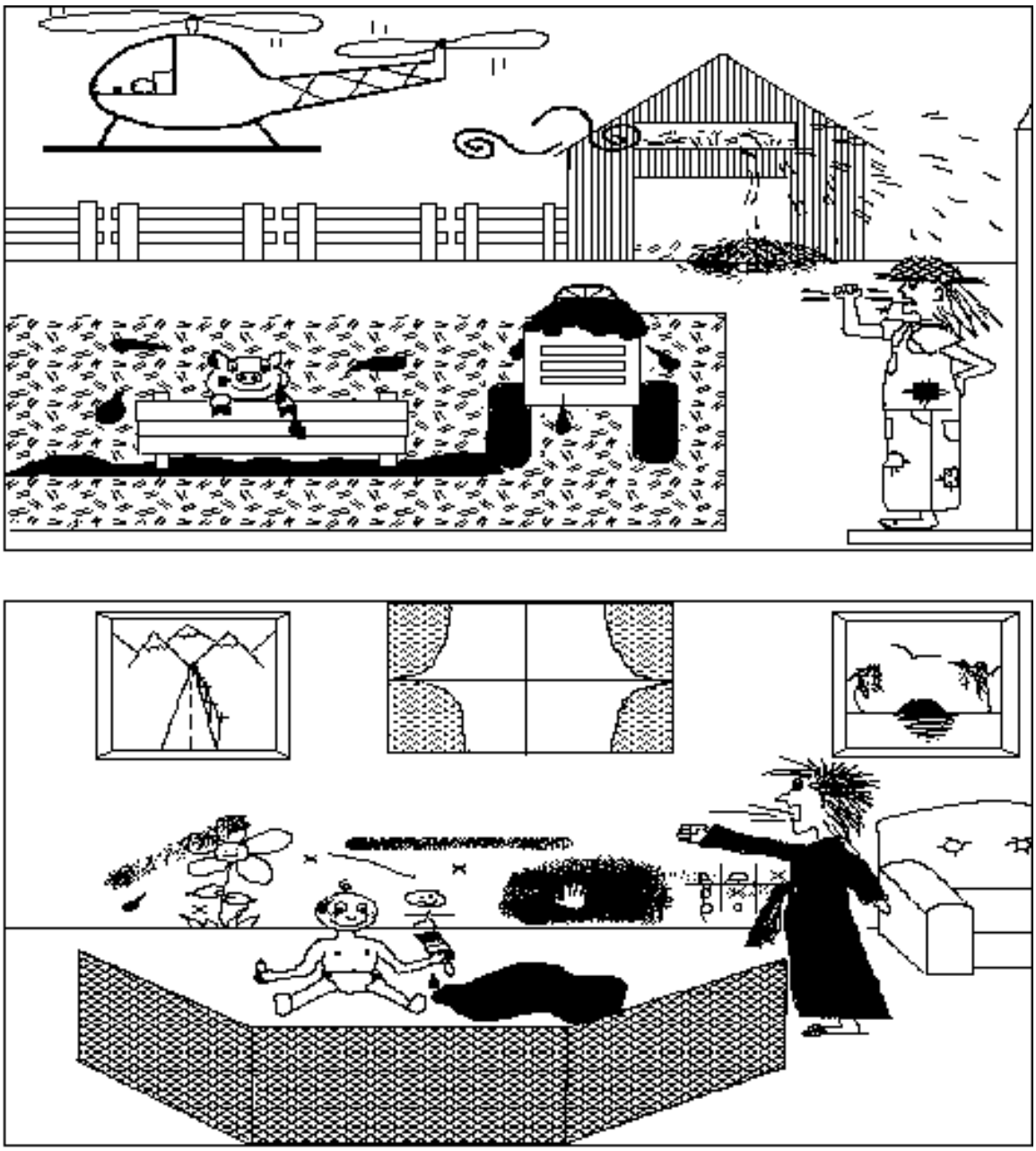

Figure 1. Sample scene pair from the Comparison Task. The top scene is the *target* and the bottom the *base*. The pig and farmer in the target scene are alignable with the baby and mother, respectively, in the base scene. The helicopter in the target is nonalignable with the base scene

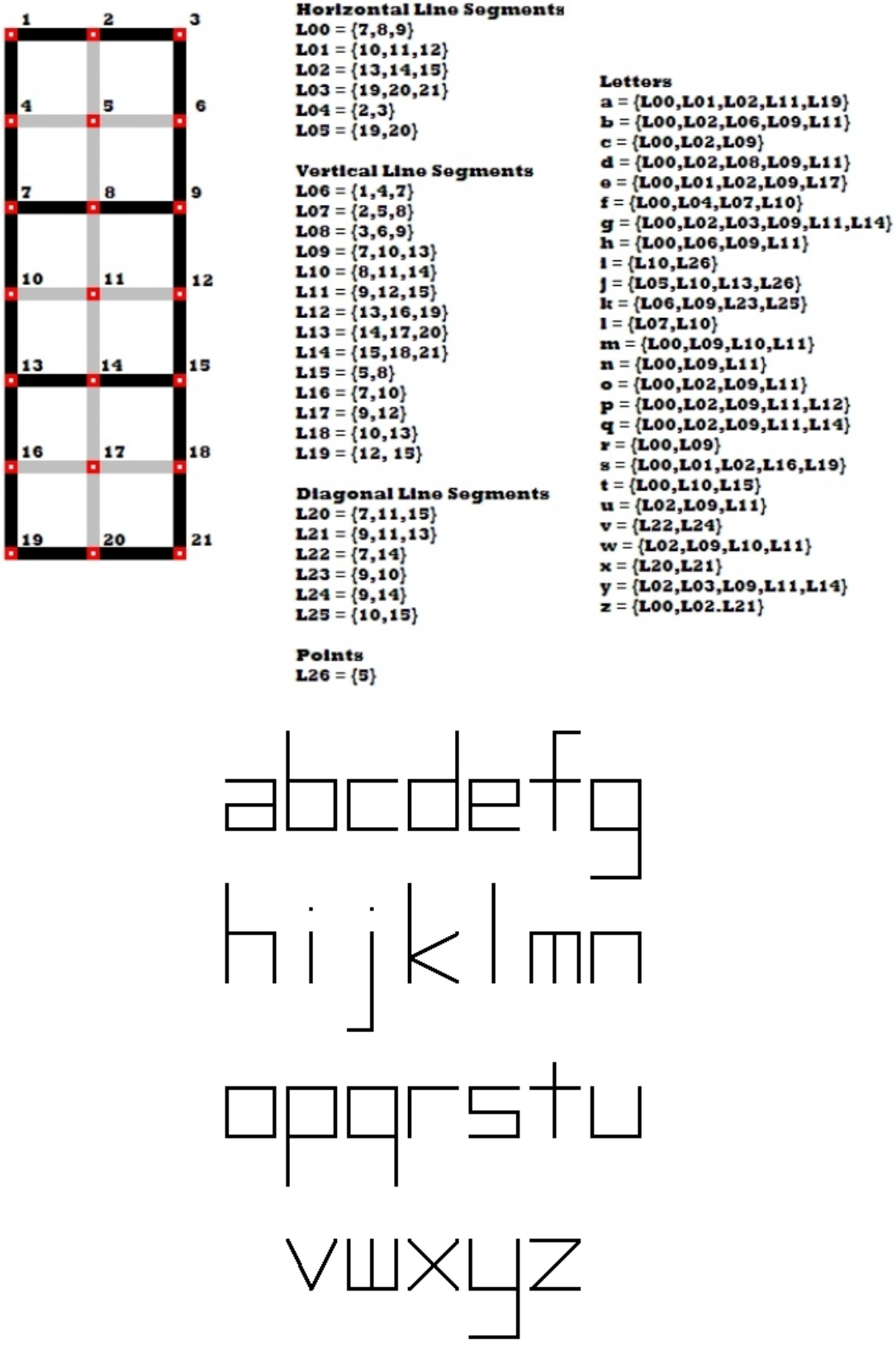

Figure 2. STRANG's letter grid and letters.