Computational Analogy
An Algorithm for Detecting Analogies

Ken Wellsch
Marlene Jones

Logic Programming and AI Group
Department of Computer Science
University of Waterloo
Waterloo, Ontario
N2L 3G1

Research Report CS-87-01
January 1987

Computational Analogy
An Algorithm for Detecting Analogies

Ken Wellsch and Marlene Jones

Logic Programming and Artificial Intelligence Group
Department of Computer Science
University of Waterloo
Waterloo, Ontario
N2L 3G1, CANADA

ABSTRACT

Evidence suggests that analogy is a key component in human reasoning and learning situations. Analogy is both a common reasoning and instructional technique. We briefly review research, beginning with the 1950's, up to the present, and in greater detail the work of Burstein, Gentner and Winston. We present a particularly simple (the computational complexity is polynomial) but powerful algorithm for detecting analogies. We examine the effectiveness of this algorithm, which is based on subtree matching, as a means of detecting analogies within two strikingly different contexts: two-dimensional scenes, and instructional systems (such as ICAI systems). Two-dimensional scene analysis is a domain rich in analogies that are also intuitively verifiable; this domain is used for our first tests. There are several reasons why within an instructional or diagnostic system one would want to detect analogies; these are discussed herein. As an initial study in this regard, a particular framework for student modelling — the genetic graph — and a variety of examples from the domain of elementary ballet are employed as our second set of tests. The results are discussed along with suggestions for future research.

Computational Analogy
An Algorithm for Detecting Analogies

Ken Wellsch and Marlene Jones

Logic Programming and Artificial Intelligence Group
Department of Computer Science
University of Waterloo
Waterloo, Ontario
N2L 3G1, CANADA

1. Introduction

One of the problems with the the notion of analogy is understanding exactly what it is. At the most basic level, *analogy is a relation of likeness between two things*. These *things* may be objects, relations or complex concepts, taken from either the same domain or disparate domains. One dominant characteristic of analogy is that there often appears to be no apparent relationship between the two *things*. On a more complex level, analogy manifests itself in the literary world as *metaphors* and *similes*. For example, consider the metaphor:

> *"What light from yonder window breaks?*
> *It is the east, and Juliet is the sun! ..."*

A second example, illustrating the lack of an obvious relationship between the *objects* is (Quine and Ullian 1978):

> *"knowledge in some ways is like a good golf score ..."*

The complete statement removes the mystery:

> *"knowledge in some ways is like a good golf score: each is substantially the fruit of something else, and there are no magic shortcuts to either"*

One of the meanings of the Greek word "analogia", from which analogy originates, is *proportion* (Polya 1954). The notion of proportion appears in a problem-solving environment; a common example are intelligence-tests. A proportion problem often has the form "A:B::C:?", i.e. A is to B as C is to what? The familiar geometric-analogy intelligence-test problem is an example of a proportion problem using two-dimensional shapes for A , B , and C .

As far as terminology goes, the structure of an analogy is divided into two parts: the *base* (or *vehicle*) and the *target* (or *topic*, traditionally called the *tenor*). The base of an analogy is the domain with which the reader is assumed to be familiar (the *base* from which to infer). The target of an analogy is the domain of which the reader has little or no knowledge (the *target* of the inferences). Analogy in teaching and communication is used to impart a better foundation for understanding new domains (of which little or nothing is known by the listener) in relation to a domain familiar to the listener.

Good analogies represent extremely efficient (compact yet rich) representations of knowledge. For example:

"The hydrogen atom is like the solar system"

Here, it is assumed the reader has some knowledge of the "solar system" (the *base* of the analogy) and that certain (but not all) characteristics of our solar system (planets revolve around the sun, the sun is significantly more massive than any planet) have corresponding characteristics in the "hydrogen atom" (the *target* of the analogy). The relationship between atom and solar system may not be obvious (as in out last example), but the characteristics that electrons (like planets) revolve around the nucleus (like our sun), and that the nucleus is significantly more massive than an electron, are represented. If we were to also infer that the nucleus "was hot" and that people live on one of those electrons, we would be extending the analogy to far. Rarely can one make exhaustive inferences from an analogy; more often, only a select few are intended.

2. Background

The study of analogy is a modern endeavor, although the use of analogy is old. The diversity in existing research on analogy makes it difficult to to fit the area into set categories; hence, a chronological ordering for the discussion of analogy research will be used. We will first discuss briefly a wide range of research, starting in the 1950's, up until the present, from both the fields of Psychology and Artificial Intelligence. This overview will then be followed by a more detailed look at the work of three people who's research is closely related to section 3.

2.1. The 1950's

In the later part of the 1950's, two prominent scientists independently gave credit to the significance of analogical reasoning in past scientific achievements and that analogy was a key component in scientific theory formation and development. Polya, a mathematician (Polya 1954), noted for his work on mathematical problem-solving, and Oppenheimer, a physicist (Oppenheimer 1956), discussed the historical significance of analogy in their respective fields, thus beginning a period of analogical research that studied the use of analogy in famous theories and its use in scientific theory.

2.2. The 1960's

The "History of Analogy in Scientific Reasoning" theme continued into the 60's; the extensive work of Hesse looked at the use and validity of models and analogies in science (Hesse 1966), and the work of Dreistadt began a long tradition of analogy classification that has been continued by later researchers (Dreistadt 1968). Unfortunately, even today, these classifications remain almost as diverse as the area itself. The work of Evans in the early 60's was a dramatic departure from the dominant study of the day (Evans 1968). For his Ph.D., Evans developed a LISP program capable of solving high-school geometric analogy intelligence test problems. The representation and decomposition of two-dimensional pictures were major problems in themselves; unfortunately Evans did not attempt to model the human process for solving such problems, reducing slightly his contribution to the study of analogical reasoning.

2.3. The 1970's

The 1970's saw a substantial increase in research on analogy; largely in psychology. Psychologists are primarily concerned with studying how people recognize and use analogies and began to develop their first models for analogical reasoning. The earliest model, introduced by Rumelhart and Abrahamson, was based on representing objects in a Euclidean space, indexed by a select number of features (Rumelhart and Abrahamson 1973). The similarity (a relative of analogy) between two objects, is simply the Euclidean distance between them. Close proximity in this feature space implied greater similarity. The model has been largely criticized: it is not completely general nor sufficiently rich to encode all

types of semantic relation (made by Rumelhart and Abrahamson themselves), zero distance (for identity) is not realistic, a distance function is symmetric while many metaphors are not, and the triangle inequality does not hold as it should (Tversky 1977). Lastly, selecting the features to use for indexing such a space and the assignment of the actual spacial positions are both nontrivial problems. Along with criticizing the simple Euclidean distance function, Tversky contributed a great deal towards developing a more sophisticated model for judging similarity (Tversky 1977), his that attempted to overcome these problems and others. Ortony extended Tversky's research by looking in more detail at knowledge representation and context/salience within an analogy (Ortony 1979). He suggests that attribute inequality arises because an analogy is a matching at a higher level of abstraction in some assumed taxonomic structure.

On a different tact, Reed, Ernst, and Banerji (1974), looked at a specific form of analogy, analogical reasoning in state-based problems (e.g. "Missionary-Cannibals Problem"). Their most interesting finding was that students (in the several experiments they administered) had better success solving two successive state-based problems of a "similar" nature when presented with the "more specific" problem followed by the "more general" problem rather than the converse ordering. This suggests that apparently the students could recognize a generalization more readily than they could a more specific instance of a problem.

Sternberg looked at a third area of analogy, that of the proportion problem (Sternberg 1977). He developed a model for proportion problems based on three operations: *inference*, *mapping* and *application*, and discusses existing theories of the day in terms of these three operations. Probably his most crucial contribution relates to whether the three operations are done *exhaustively* or are *self-terminating*. Exhaustive means the operation is applied to all data, while a self-terminating operation applies some test to each datum as it it produced. Through administering several experiments, Sternberg concluded that mapping and application were probably self-terminating and had insufficient evidence whether inference was exhaustive or self-terminating.

By the late 70's the *schema* developed as a popular memory model. Schustack and Anderson (1979) were among several researchers who studied schema formation and the effects of analogy (past experiences) on the development of new schemas through various experiments. Of note was their conclusion that the process of using past experience seemed to be independent of the abstract-directness of the material. Cues to both specific knowledge and general knowledge both improved problem-solving performance.

Research that incorporated some form of analogy in a computer application remained rare unlike the boom in Psychological research. Two examples are of note: the work by Kling (1971), and the work by Moore and Newell (1974). Kling's research involved first-order theorem proving and the connection with analogy was the introduction of techniques to exploit past proofs deemed to be sufficiently similar. The incredible potential of analogy in this regard was certainly recognized before the early 70's but the research on analogy had not progressed sufficiently to allow the introduction of such techniques until the 70's. Although it was a major accomplishment of the time, it relied extensively on the user who had to represent the past knowledge and direct the theorem prover to the most appropriate past problem and solution. Moore and Newell are well-known for their program called **Merlin**, a program originally intended to understand Artificial Intelligence but later restricted to just programs in that domain. One operation provided by **Merlin** was the mapping operator (/) that could be used to ask whether an object X could be viewed as another Y (as in X/Y). They found something that is better understood today: the more general the method (as in mapping), the less powerful it becomes compared to domain specific techniques.

2.4. The 1980's

With the coming of the 1980's, the volume of research by Artificial Intelligence people has grown almost as dramatically as research in psychology did in the 70's. Analogy receives credit in several applications programs, and the distinction between the human models and the applications models has largely vanished. The volume of research may have reached higher proportions, but it still remains diverse: Mulholland, Pellegrino, and Glaser (1980) studied the encoding phase of Sternberg's model to discover that increased object complexity in an analogy did not increase error rates while an increase in transformational complexity did increase error rates. The limited human short-term memory was given credit for this disparity. Rumelhart and Norman (1981) continued the schema theme started by Schustack and others. They, and even more so by Halasz and Moran (1982), discuss the danger of analogy; analogies can even be harmful since there are only a select number of inferences contained within an analogy that are intended while the many other potential inferences can be dangerously wrong. Tourangeau and Sternberg (1981) briefly revived the feature-space/distance-function model of analogy in the context of Ortony's work on similarity by distinguishing within-domain distance and between-domain distance only to achieve inconclusive and disappointing results with human subject experiments. Gick and Holyoak studied human analogical reasoning (1980,1983) using story problems rather than the traditional proportion problems, that are undeniably more artificial. They studied, using several well-described experiments, how the use of one or more analogous problem/solutions could effect later problem solving in student subjects. Their conclusions are interesting: they found that knowing of a single analogous problem/solution produced remarkable success rates in solving later like-problems, and that the same effect occurred only when presented with several analogous problem/solutions when the connection was not made explicit.

Equally diverse are the various uses of analogy within a variety of applications (implemented and suggested): Langley, Bradshaw, and Simon (1981) reference the effectiveness of analogy in their program *BACON*, a system intended to discover elementary laws from empirical data. They employed past solutions to reduce the amount of redundant computation, and at a higher level when a new law utilized terms similar to a previous law, characteristics within the previous law where used to reduce the volume of data examined. Lenat (1982) credits analogy as the most effective method for generating new heuristics over generalization and specialization. Carbonell's theoretic work (1981) on the use of past experience in problem solving uncovered the now obvious combinatorial explosion problem that exists in the use of analogy in problem solving. He found that just remembering the problem and solution was not sufficient, that the solution derivation was also necessary. This leads to a problem with representation and complexity owing to the large volume of information that must be maintained. Unfortunately Carbonell does not deal with representation in sufficient detail. Along with Minton (1983), Carbonell also considers the common-sense reasoning problem based on metaphor comprehension. Their model is based on the belief that there exists only a moderate number of general metaphor forms that can be used as templates for reasoning. Unfortunately the feasibility of this idea is difficult to determine due to insufficient detail.

Recent research has been directed at formalizing analogical inference and mapping constraints. Kedar-Cabelli (1985) has proposed the specialized notion of *purpose-directed* analogy. Given a particular task (or goal) in a problem-solving environment, use the underlying purpose to constrain the mapping phase. Kedar-Cabelli proposes an explicit representation of the *'purpose of an analogy'*. What remains is a technique for deriving the purpose of a task automatically. One may question whether it would be simpler to provide the relevant mapping to begin with, rather than an analogy, but this may be impossible when the reasoner cannot expect external assistance (as in an independent robot). Indurkhya (1985) presents a formal theory of analogical reasoning based on first-order predicate calculus, called Schema Language (SL). On top of this language, Indurkhya developed two domain-independent theories called Constrained Semantic Transference (CST) and Approximate Semantic Transference (AST). *Consistency* is used as the central constraint for analogical mapping. Greiner (1985) also introduces a theory of analogical reasoning using a formal basis. Greiner's notion of analogical inference is similar to that of Indurkhya's, although Greiner introduces stronger constraints based on the use of *abstractions*. Both

Greiner and Indurkhya have set the stage for further research on formal theories of analogical reasoning.

2.5. Fundamental Research (Current)

Gentner is a researcher in Cognitive Psychology who has published several works in the 80's that defines the current model for analogy. Her *structure-mapping* theory of analogy provides a set of principles for the representation and derivation of analogies. Here is the definition of structure-mapping (Gentner 1982):

attributes – predicates taking one argument,
relations – predicates taking two or more arguments.

T *target* containing object nodes $t_1, t_2, \cdots t_m$
B *base* containing object nodes $b_1, b_2, \cdots b_n$

A , R , and R' *predicates*

Example :

"The hydrogen atom is like the solar system"

Predicates:

sun (hot), sun (massive)
revolves-around (planets , sun)
more-massive-than (sun , planets)
more-massive-than (nucleus , electron)

Map object nodes of B onto object nodes of T

M: $b_i \rightarrow t_i$

(i.e. sun $\rightarrow$ nucleus , planet $\rightarrow$ electron)

Using node substitutions found above:

M: $[R\ (b_i, b_j)] \rightarrow [R\ (t_i, t_j)]$

where $R\ (b_i, b_j)$ is a relation that holds in domain B .
and so on for higher order predicates...

(i.e. revolves-around(planets,sun) $\rightarrow$ revolves-around(electron,nucleus))

A distinguishing characteristic of analogy (so claims Gentner) is that attributes from B tend not to be mapped into T :

M: $[A\ (b_i)] \not\rightarrow [A\ (t_i)]$

(i.e. sun (hot) $\not\rightarrow$ nucleus (hot)).

In addition to the structure mapping theory, Gentner adds the following principle:

> *" ... Systematicity Principle: A predicate that belongs to a mappable system of mutually interconnecting relationships is more likely to be imported into the target than is an isolated predicate."* (Gentner 1982)

Winston has done a wide range of work that, to varying degrees, has incorporated analogy. One paper in particular (Winston 1980) is devoted to the study of learning by analogy via a computational model. Winston deals mainly with representing story plots (such as *Romeo and Juliet*, and *Macbeth*) and the determination of some measure of plot similarity. For example, the play *Romeo and Juliet*, and *Cinderella* have a lot in common. These stories are represented via a network of frames (i.e a graph), and using a constrained matcher, a value representing the number of similar components is determined. An

exhaustive matcher would be far too computationally expensive, so Winston selected a set of constraints to limit the combinatorial explosion; these constraints include: *cause*, *importance*, and *classification*. One criticism of Winston's approach is his reliance on well-bounded domains. The story plot is clearly a self contained world about the story. A point worth noting is that Winston is looking at generalizing the target and base (i.e. schema induction) rather than transferring specific relations using some (supposed) similarity between the domains. His approach is different from those people studying analogical reasoning; his task is somewhat easier, relying on some external source for accurate representation of the domain knowledge.

Burstein takes a different approach. He is interested in the reasoning aspect; to extend incrementally a small target domain with the aid of an external teacher and some knowledge about the base domain (Burstein 1983). The method he employs is less clear than Winston's but based on the structure-mapping model of Gentner. Burstein criticizes the well-defined knowledge necessary for the work of Evans and Winston as well as the complexity of a partial-pattern matching approach. Burstein suggests a hierarchical knowledge structure to reduce the volume of potential pairings. He also deals with non-identical mappings, something he claims is not embodied in Gentner's model. A *virtual* relation is created when a non-identical mapping occurs to preserve the other relationships in the mapping. Burstein employs an incremental process of mapping so that any potential errors can be recognized and corrected by the external instructor before making any further inferences. Work such as Winston's is exhaustive; the matcher goes through all of the data before returning any result.

2.6. Discussion

Research has come a long way from looking at analogy as just a scientific reasoning phenomena. Analogical reasoning has been popular for many years as an accurate measure of human intelligence. Computer programs that can solve these problems have been around for some time as well, but the question remained, how do people use analogy? Simply writing an effective geometric-analogy intelligence-test problem solver did not sufficiently address this question. Through the 70's and up until today, researchers have been struggling with this problem. A well accepted theory today is that analogical reasoning and the use of past experience can be a powerful processing tool. Cognitive Psychologists still work at better models of how people use analogy, and researchers in Artificial Intelligence use these models and their own to harness some of this power for use in various systems, such as Expert Systems. This research is still far from complete, many areas remain largely untouched. It would seem without question, that analogy must play a large role in any Learning System that really does learn and function with *intelligent behaviour*.

It is evident from the literature that research is as diverse and ill-defined as the notion of analogy itself. The lack of a clear definition makes analogy a vast problem. The domains encompassed by analogy range from those in which precise mappings operate well, to domains in which even representation of the knowledge is highly complex.

3. The Algorithm

The problem of detecting and applying analogies is very hard and although a great deal of work has been directed at analogical reasoning, it remains an open problem (for a fuller discussion of analogy and analogical reasoning, see (Wellsch 1985)). In the context of this paper, the subproblem of detecting analogies is tackled using a simple, but powerful polynomial algorithm (Wellsch 1986b).

A method for modelling analogy consists of **matching** two bodies of knowledge (base and target) obtaining pairwise correspondences between the two bodies (in order to determine the analogy). The analogy (these pairwise correspondences) can be used to **map** relationships known to hold in the base domain to the target domain (i.e. analogical reasoning).

If one accepts this matching paradigm for modelling analogical reasoning, then several key issues remain to be formalized. An effective matching procedure and mapping procedure are two obvious issues. A third, equally important issue is the form of knowledge representation (KR) to be used by

these two procedures. A proportion of existing analogy research has dealt with matching and mapping trying to remain above actually selecting a KR form; unfortunately the computational feasibility of any approach relies on selecting a form of KR and providing concrete procedures. The algorithms that provide matching and mapping rely on the KR chosen.

3.1. Knowledge Representation

The array of potential KR forms (frames, networks, and logic for example), all revolve around one basic (fundamental) representation structure: a **graph**. A **tree** and a **list** are restricted forms of graphs that are also used when the complexity associated with graphs is unnecessary and undesirable. The disadvantage of an unrestricted graph in a matching situation is well-known: the *subgraph isomorphism problem* is *NP complete* (Gary and Johnson 1979) i.e. graph matching is computationally expensive and potentially infeasible. Conversely, restricted forms of graphs are more computationally attractive, for example, *the tree isomorphism problem* has a linear-time solution (Aho, Hopcroft and Ullman 1974).

In addition to computational superiority, a hierarchical representation has a basis in the current Psychological research on knowledge structuring (Bourne, Dominowski and Loftus 1979; Dember and Warm 1979; Reynolds and Flagg 1977). The fact that people have a limited short-term memory capacity has lead to the *chunking* model for short-term memory (Miller 1956). A limit of 7 ± 2 *chunks* of information as the human short-term memory capacity is well accepted and experimentally verified. For people to remember more than such a restricted number of specific things requires some way of encapsulating the surplus within each *chunk*. The theory of *chunking* encorporates an explicit hierarchy by *chunking chunks* to form larger and large units of information. Hierarchical recoding of information is a common component in cognitive models for human memory; a tree structure provides such a hierarchical representation.

A hierarchical representation is also used in Pattern Recognition. Scene and picture analysis are two important applications of pattern recognition (and analogies in these visual domains are common in human reasoning). Syntactic pattern recognition, one of the two general pattern recognition approaches (Fu 1982), represents two and three-dimensional scenes using a hierarchical description.

It would seem that a hierarchical representation (without cycles and a specific rooted node) is an attractive knowledge structure. The question of descriptive power immediately arises; do trees have sufficient descriptive power compared with graphs? The most obvious disadvantage of a hierarchical structuring is the problem of multiple representation. By this we mean that a given scene can be "ordered" in several different ways within a tree. Gestalt psychology does provide what amounts to heuristics that describe how people mentally order scene hierarchies, but these "guides" have not been tested in this research area. This is one of the issues we will address herein.

3.2. Matching

Tree matching is an extension of the tree isomorphism problem. When a tree is used to represent *knowledge*, the nodes of the tree correspond to objects and the relationships between the objects (i.e. the tree has labelled nodes). The tree isomorphism problem requires that the structure and labels be identical between the two trees (with branch permutations) for them to be isomorphic; a strict form of equality. On the other hand, an exact match between the base and target of an analogy (assuming one would call such a comparison an *analogy*) is of little value. To deviate from strict equality (to flexible equality) relies on some additional knowledge about the labels. Clearly, if our model of equality relies on labels being identical for equality and otherwise being not equal, then our model of analogy could not function. The background information regarding labels provides a means of judging the *similarity* of two labels (not just "equal" or "not equal"); strict equality is inadequate in this situation.

The area of judging similarity has received a reasonable amount of attention (Ortony 1979; Tversky 1977); the method used here does not attempt to achieve the same level of sophistication of today's similarity models, but has proved adequate for developmental purposes. The relationships between the various tree labels of the domain trees are represented in a single hierarchy called a *background*

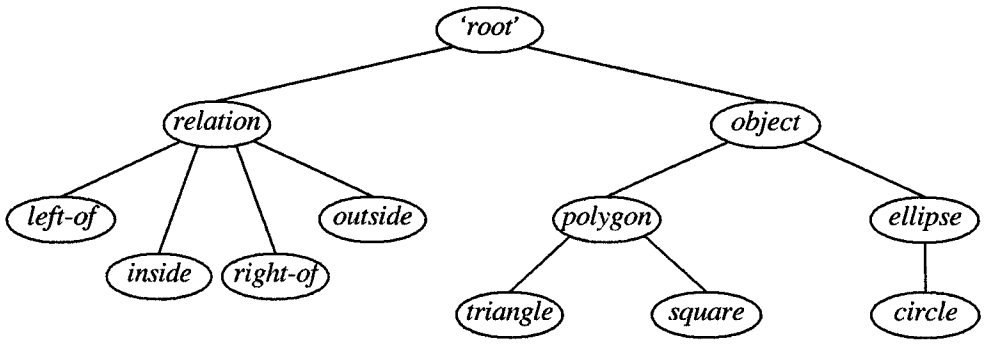

Figure 1
General background knowledge hierarchy (BKT)

knowledge tree (BKT) (see Figure 1). The *similarity function* σ, given two labels, computes a value from the relative positions of the two labels within the BKT, i.e. a measure of distance. Greater "distance" implies lower similarity between the two labels. This form of measure is certainly not ideal, but whatever metric is chosen, as long as it yields values such that a larger value signifies lower similarity, then the exact definition of σ will not effect the matching algorithm.

Returning once again to the idea of tree matching, given the function σ, one can develop an algorithm of "flexible" matching for trees. There are a variety of ways of combining the values of individual node pairings (obtained from σ) to judge the similarity of two trees. For example, the worst node-pair rating, the best node-pair rating, and the sum of all the node-pair ratings (cost) are three such measures; others certainly exist. In the research discussed here, the worst node-pair rating and the total "cost" are used to judge tree similarity. Matching two trees t_1 and t_2 yields the ordered pair $<cost\ (t_1,t_2), worst\ (t_1,t_2)>$. Notice that the *cost* function pairs the most appropriate (least costly) branches, and the overall cost is penalized for any remaining unmatched branches (the *cost* function presented here assumes, without loss of generality, that $|t_1| \leq |t_2|$).

Consider now the subtree matching problem: given two trees, t_1 and t_2, is one tree isomorphic to a subtree of the other? Note that a subtree s of a tree t can be the tree t itself (i.e. $s = t$) and that we are not considering the problem of matching **all** the subtrees of t_1 with all the subtrees of t_2 (That would be grim indeed!). Here we have a selection problem: which subtree of t_1 *best* matches with t_2 as given by the $<cost, worst>$ rating? The selection strategy used is one that picks the matching with the lowest worst node-pair rating (i.e minimize worst node-pair rating), and if there is more than one that has this minimum rating, then break such a tie by minimizing the cost rating among the minimum worst node-pair matchings (i.e pick the matching with the lowest cost rating). Subtree matching consists of selecting the "best" subtree pairing.

Finally, at the level of the base and target representation, is *forest* matching. Both the base and target domains are represented by forests, i.e. sets of trees. It is at this level of representation that we refer to matching as that associated with analogy. Forest matching uses the same basic approach taken with subtree matching (i.e. the selection technique) but with a few twists. Things become a bit more complicated because the trees in a domain forest are significantly more independent than the nodes in a tree (other than pertaining to the same domain, there isn't necessarily any more of a connection). This structure independence (and common-sense) implies that combining an unequal number of structures (trees) taken from one domain with another has no basis. The natural approach might be to simply eliminate each of the already paired trees (obtained from "best" matching) from consideration and use only the remainder for later pairings (i.e. a process of elimination).

$$cost\ (t_1,t_2) \equiv \textbf{if}\ root\ (t_1) = root\ (t_2) = \textbf{nil then}\ 0$$
$$\textbf{else if}\ \delta(root\ (t_1)) \neq \delta(root\ (t_2))\ \textbf{then}\ \text{LARGE-CONSTANT} \times (\,|t_1| + |t_2|\,)$$
$$\textbf{else}\ \sigma(root\ (t_1),root\ (t_2)) + \sum_{i=1}^{\delta(root\ (t_1))} cost\ (child\ (i\ ,t_1),child\ (i\ ,t_2))\ \textbf{endif}.$$

$$worst\ (t_1,t_2) \equiv \textbf{if}\ root\ (t_1) = root\ (t_2) = \textbf{nil then}\ 0$$
$$\textbf{else if}\ \delta(root\ (t_1)) \neq \delta(root\ (t_2))\ \textbf{then}\ \text{LARGE-CONSTANT}$$
$$\textbf{else}\ \max_{i=1}^{\delta(root\ (t_1))} \left\{ \sigma(root\ (t_1),root\ (t_2))\ ,\ worst\ (child\ (i\ ,t_1),child\ (i\ ,t_2)) \right\}\ \textbf{endif}.$$

$$pair\text{-}min\ (\langle cost_1,worst_1\rangle,\langle cost_2,worst_2\rangle) \equiv \textbf{if}\ (worst_1 < worst_2)\ \textbf{or}$$
$$((worst_1 = worst_2)\ \textbf{and}\ (cost_1 < cost_2))\ \textbf{then}\ \langle cost_1,worst_1\rangle$$
$$\textbf{else}\ \langle cost_2,worst_2\rangle\ \textbf{endif}.$$

$$\sigma\ (l_1,l_2) \equiv\ \textbf{if}\ inconsistent\ (l_1,l_2)\ \textbf{then}\ \text{LARGE-CONSTANT}$$
$$\textbf{else}\ (\,|\,depth\ (l_1)-depth\ (l_2)\,|\ +\ \textbf{MAX}(\,|\,depth\ (l_1)-depth\ (\rho)\,|\,,$$
$$|\,depth\ (l_2)-depth\ (\rho)\,|\,))\times\textbf{MAX}(priority\ (l_1),priority\ (l_2))\ \textbf{endif}.$$

where

depth (n) is the depth in the background knowledge hierarchy of node n and ρ is the common ancestor of l_1 and l_2.

$$tree\text{-}match\ (t_1,t_2) \equiv \langle cost\ (t_1,t_2),worst\ (t_1,t_2)\rangle.$$

$$subtree\text{-}match\ (t_1,t_2) \equiv \underset{t\ \text{subtree of}\ t_1}{pair\text{-}min} \left\{ tree\text{-}match\ (t\ ,t_2) \right\}.$$

$$forest\text{-}match\ (f_1,f_2) \equiv \underset{t \in f_1}{pair\text{-}min} \left\{ \underset{\text{eliminate}\ t_j\ \text{from}\ f_2}{subtree\text{-}match}\ (t\ ,t_i)\ \mid\ t_i \in f_2 \right\}.$$

Figure 2
Matching algorithm

Two obvious problems can occur using such a matching algorithm. The first is *multiplicity*, i.e. a one-to-many mapping of an object or relation to another. In terms of a tree representation, this manifests itself when a single leaf node is paired to a subtree composed of more than one node. The second problem is *inconsistency*, i.e. a mapping that contains an object or relation that seems to map to more than one distinct object or relation (or visa versa). This is much like multiplicity, but simpler to deal with than multiplicity. The algorithm penalizes a matching that results in inconsistent mappings (in function σ). The algorithm (Figure 2) handles both of these problems (although there are undoubtedly other approaches).

3.3. Mapping

The mapping process is straight-forward given the node-wise pairings obtained in the matching stage. In most cases, the direction of the mapping is from the base domain to the target domain (a characteristic of analogy). The mapping stage can be carried out by simply **substituting** the label names in the base structures by their paired label names from the target structures. In this way the base structures can be *looked upon in terms of the target domain.* One question that immediately arises is: "What happens when there are labels in the base structures that have no correspondence in the target structures as determined by the matcher?". A second question relates to consistency: "What happens when there is more than one possible substitution for a given label, i.e. a non one-to-one mapping of labels?". Figure 3 illustrates two examples that contain potential mapping conflicts.

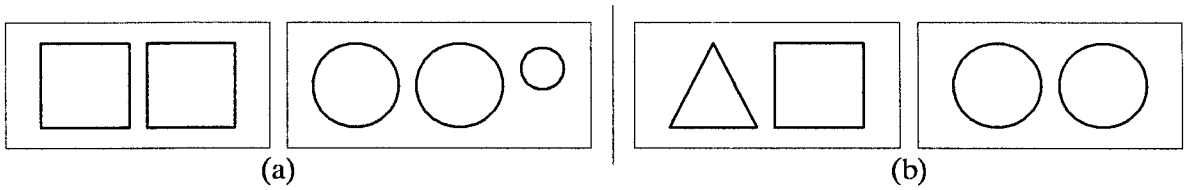

Figure 3
Pictures with mapping conflicts

Regarding the first question, these unpaired labels may be objects or relations that have some equivalence in the target domain but the reasoner does not have sufficient knowledge of the target domain. It is also possible that such objects and relations do not have any correspondence in the target domain. Here is where analogical reasoning takes on its full characteristics. Analogies are prone to causing reasoning errors (lead to false conclusions); this is an accepted characteristic. Analogy is not a proof technique, but merely a powerful guide. Thus it is necessary that some form of *alias* be established for these unpaired objects and relations so that at some later time their existence in the target domain can be tested and either verified, discounted, or left as still unknown.

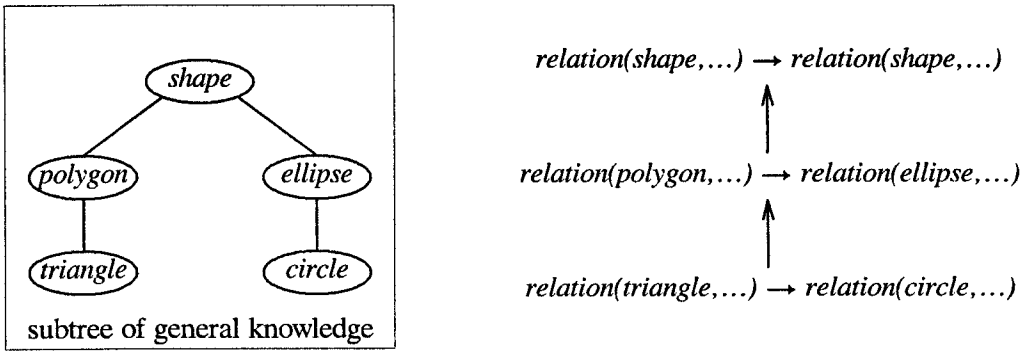

Figure 4
Range of generalized substitutions

As a means of dealing with the unpaired labels, the choice of determining a **mapping union** or a **mapping intersection** is available. *Union* consists of ignoring the lack of a pairing and just directly transferring the label into the target domain. On the other hand, the *intersection* eliminates any labels that are not integral within the portion of the structure that contains paired labels. For example, if a subtree contains paired labels and the root label has no pairing, then the root label must has some correspondence within the target domain. In such a case, an *alias* is established for the label.

Regarding the question of consistency, either the matching algorithm must take it into account in the selection process, or should inconsistency remain in a matching, then some other strategy must be followed by the mapping stage (such as ignoring the inconsistent pairings all together). It is straightforward to extend the node matching function to consult a list being maintained by the tree matching algorithm to see if the node pairing in question is consistent with the preceding node pairings in the match. If it is not, some large value akin to (LARGE-CONSTANT) can be assigned the pairing to indicate the inconsistency.

One interesting extension to the simple substitution algorithm for mapping is a **generalized mapping** (see Figure 4). Such a mapping consists of replacing the *most-specific* labels paired by the matcher with successive generalizations obtained from the background knowledge tree (BKT). Since the background hierarchy is based on abstraction, tracing successive generalizations until the common ancestor of the two original labels is encountered is a simple task. By augmenting the simple substitution with generalized substitutions the wealth of potentially useful reasoning structures increases. The full use and implications of these generalized mappings remains to be studied.

4. Implementation and Testing using Two-Dimensional Shapes

The algorithm described in section 3 has been implemented and tested in several domains; the major domain being two-dimensional shapes (as in geometric analogy intelligence test questions). Consider, for example, the two pictures (1) and (2) in Figure 5. Each picture is representable by a tree structure (shown for picture (1)) and can also be described by a nested predicate notation (shown for all three pictures).

Does an analogy exist between pictures (1) and (2)? Applying the algorithm in Figure 2 by pairing the root node (i.e. *inside*) of picture (2) to each node of picture (1) yields:

inside with *left-of* ☞ <(1+4×LARGE-CONSTANT+1),(LARGE-CONSTANT)>
inside with *inside* ☞ <(0+2+1),(2)>
all other pairings ☞ <(4×LARGE-CONSTANT),(LARGE-CONSTANT)>

The pairing between picture (2) and the left half of picture (1) produces a strong matching (<3,2>) over the other possible matchings. The resulting node-pairings are consistent: *inside* → *inside*, *square* → *triangle*, and *circle* → *square2*, and picture (2') is the result of the analogical inference.

Our second example (see Figure 6) is only slightly more complicated than the first; it consists of a matching that results in consistent object pairing but an inconsistent relation pairing. Intuitively, the four identical objects (the squares) are paired with four identical objects of a different form (the circles). The problem is obvious, all of the circles are positioned along the horizontal, while the last square in the left-hand picture lies below the horizontal. The two tree representations below each picture have the offending positional relation circled. The algorithm successfully matches corresponding circles with squares.

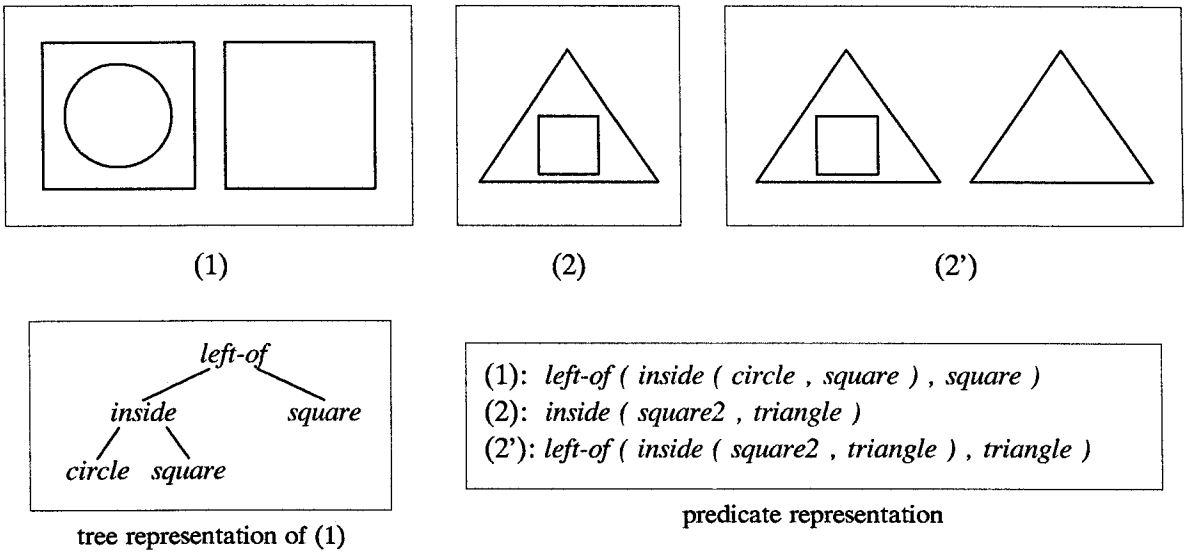

Figure 5
Two simple pictures, (1) & (2),
and resulting analogical inference (2')

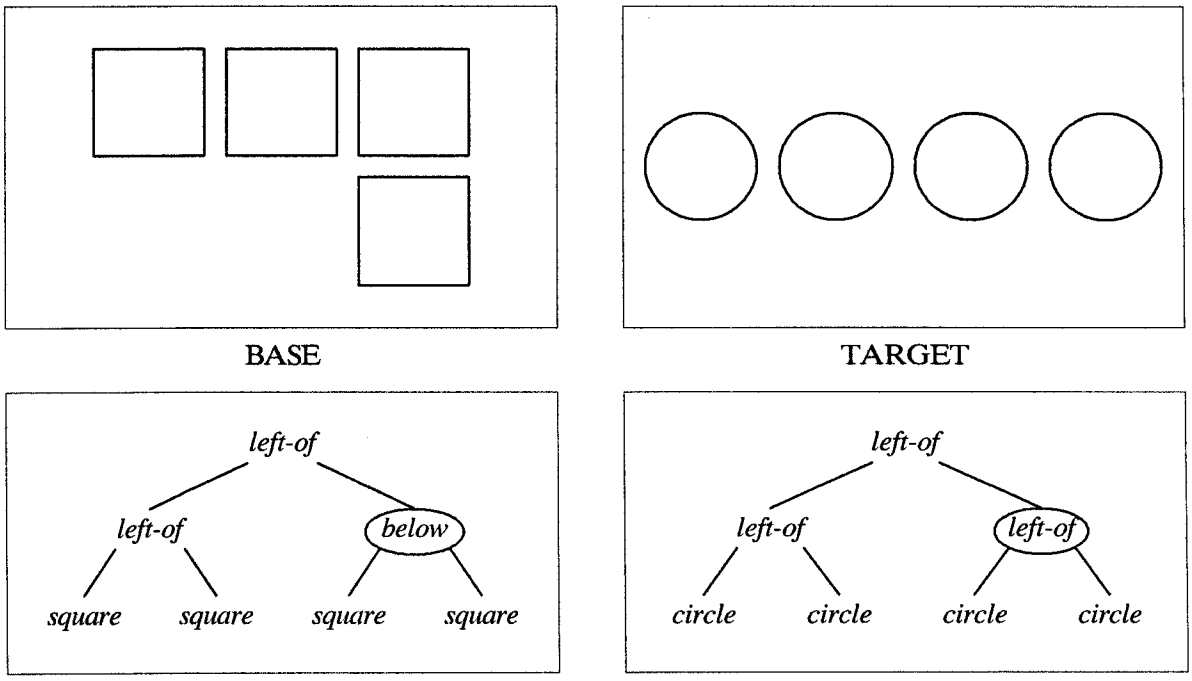

Figure 6
Single relation mismatch

This particular example also illustrates the problem of multiple representation of a scene using a tree structure. One will notice in Figure 6 that the tree representations shown group the left pair and right pair together, and then group these as a pair. Another possible ordering would be to group them from left to right, or right to left (i.e. group the left pair, then group that with the next object, and finally group that subtree with the last object on the right). Should different ordering strategies be employed for the base and target scenes, the matching usually fails to produce anything. We will discuss this problem in our concluding remarks.

Our third example is taken from a paper by Evans (Evans 1968); it represents the classic geometric analogy intelligence test problem. One interesting application for the algorithm is to group the upper A and C pictures into a forest, and the possible solution choices plus B into a second forest. The matching obtained was that A matched "best" with B and C matched "best" with 3, the solution. This remarkable result must be qualified though; in order to obtain the correct solution to the problem, it was necessary to take into account an important property of visual analogies. It has been suggested that in many analogies, relationships are more *important* than objects, and in turn, objects are more *important* than their attributes (Carbonell 1981). This is where the notion of *priority* comes from in the node similarity function σ. Relations can be given a higher priority than objects; providing this additional constraint allows for the solution of a wider array of visual analogies. The use of a *fudge factor* is not very satisfactory, but the notion of priorities and importance is not simply a *fudge factor* but a real characteristic used in human visual perception (Gestalt Psychology).

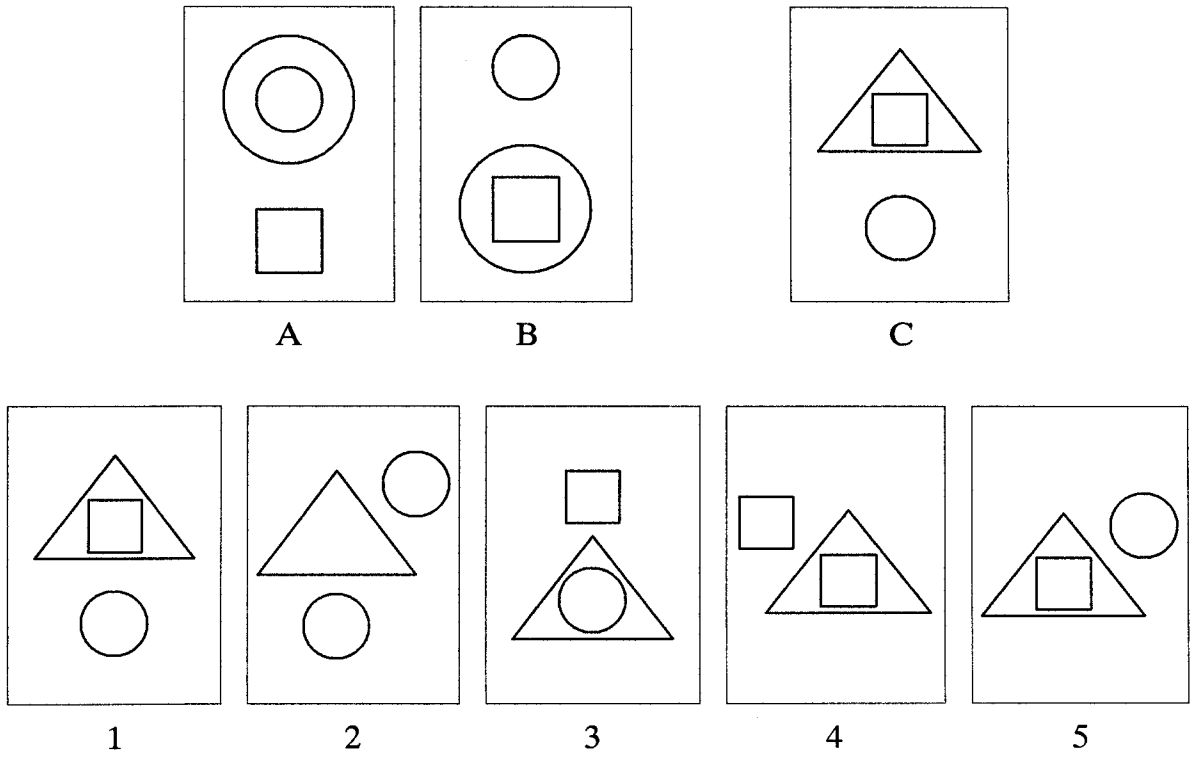

Figure 7
Geometric analogy intelligence test question (Evans, case 5)

Figure 8 is our first real look at object **multiplicity**. Multiplicity is the *multiplication* (or replication) of an object in a pairing (i.e. one-to-many mapping). The simplistic belief that all matchings will be one-to-one is unrealistic. Intuitively, the matching between the figures in the left picture have a mapping

to those in the right-hand picture. Object consistency implies that we cannot pair one circle with a triangle, and the other circle with the square; careful study shows that each pair *(triangle-square)* consistently maps to a circle. This requires that a subtree of the left representation be paired with a leaf-node (circle) of the right picture (matching multiplicity). Again, the algorithm found the forementioned solution.

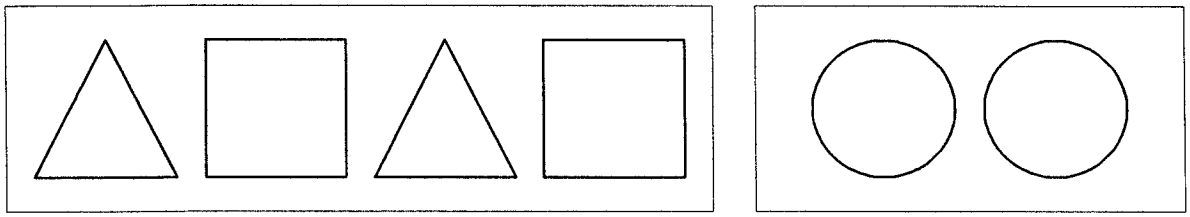

Figure 8
An example of Multiplicity

Our last example illustrates more clearly regular multiplicity (irregular multiplicity exists when the multiplication factor is not the same for each object-cluster pairing). Again, the algorithm obtained the accepted matching (*circle* → *circle* ×2, *triangle* → *triangle* ×2, etc.).

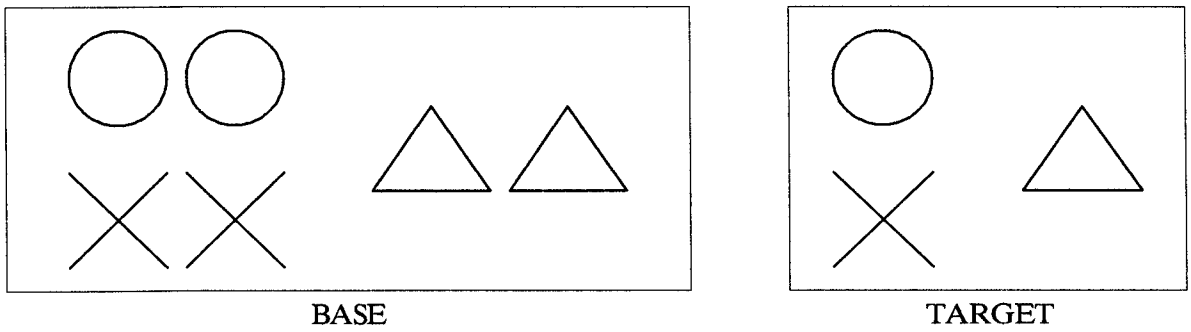

Figure 9
Example of Regular Multiplicity

5. Algorithm Adaptation to the Domain of CAI

Our second application concerns the detection of analogies within the domain of Computer-Assisted Instruction (CAI). Reasoning by analogy is a common method of learning. It appears to be a technique that humans find easy to employ, even when little data is available; in fact, it is a technique that is often employed when nothing else works. For a discussion regarding the use of analogy with instruction, see (Wellsch and Jones 1986a).

For the purposes of the discussion here, we have selected one representation scheme for a student model, the *genetic graph* that was originally proposed by Goldstein (1982). There were many reasons for selecting this particular representation scheme including its flexibility, the fact that it is has been successfully employed in a variety of domains, and the fact that Goldstein's original design included the use of *analogy* links. Although we have initially restricted ourselves to testing our algorithm within one framework, the algorithm is not constrained to this one framework and could, in fact, be adapted for use within other student model representation schemes.

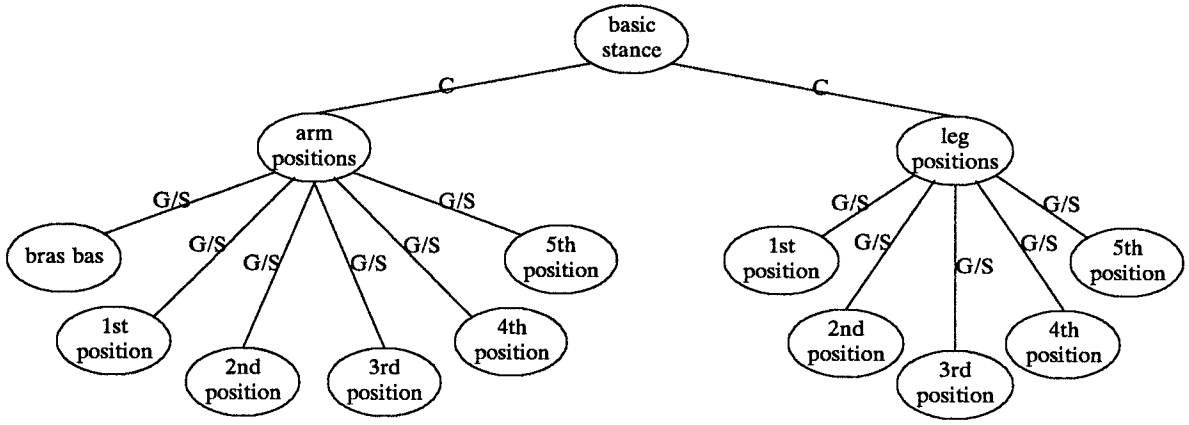

Figure 10
A subset of an introductory ballet genetic graph (Wasson 1985)

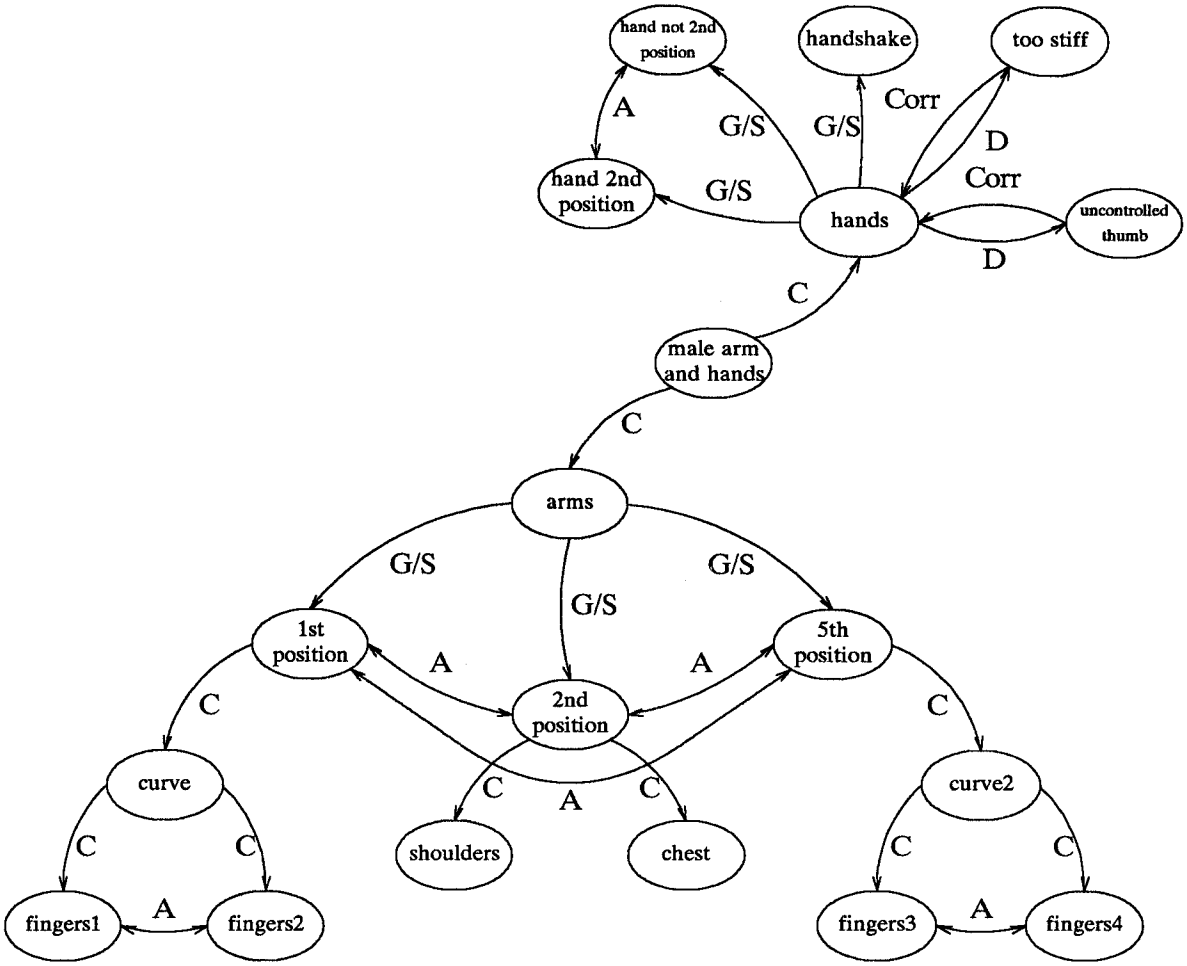

Figure 11
Another subset of an introductory
ballet genetic graph (Wasson 1985)

The *genetic graph* is a directed graph in which the nodes represent knowledge (facts, subskills and deviations thereof), the links represent relationships between the nodes (such as *generalizations, specializations, refinements, components, analogies, deviations, corrections, tests*). Moreover, the genetic graph can be viewed as a multi-dimensional structure that is divided into various *levels* of difficulty. This is a simple means of indicating the relative difficulty of subskills or concepts. Another useful means of organizing the graph's nodes is that of islands. An *island* is a cluster of nodes and the connecting links. Such a cluster may represent a single skill or body of knowledge that constitutes a concept. In order to represent both prerequisite skills and an ordering of steps within a procedural skill, *pre* and *post* links can be employed.

In other words, the genetic graph is a structure in which all desired information about the domain, deviations thereof, is stored. The student model is, in fact, an overlay on the genetic graph. One can view the genetic graph as a search space and the student model as a selection of appropriate pieces of the search space. The genetic graph approach has been successfully employed to model several diverse domains: the simple adventure game WUMPUS (Goldstein 1982), subtraction (Wasson 1985), elementary ballet (Wasson 1985), division (Dundas and Stockdale 1985). For further explanation of the genetic graph approach to student modelling, detailed examples, and a discussion of generating and maintaining the genetic graph, the reader should consult (Wasson 1985).

5.1. Representing the Genetic Graph

We wished to base the testing of our algorithm within the context of student models on previously published genetic graphs, rather than develop further examples that we might subconsciously tailor for success. Wasson's genetic graphs for the domain of elementary ballet were selected, because more information was available for these examples than those of the other previously published domains. (It is also a more interesting and challenging domain than WUMPUS, subtraction or division.) Sample genetic graphs for introductory ballet are illustrated in Figures 10 and 11. Each 'link' is labelled with a symbol that denotes its type:

C	*components*	**A**	*analogy*	**G**	*generalization*
D	*deviation*	**S**	*specialization*	**Corr**	*correction*

Figure 11 represents six basic stances or positions in ballet, partitioned into separate arm and leg positions. The *bras bas* is an arm position only, while first to fifth positions are positions for the entire body (see Figure 13).

Figure 11 highlights a decomposition based on maturity and the sex of the ballet student, the hand and arm positions for a mature male. Notice that only first, second and fifth positions are presented; this is based on the teaching approach of (Lawson 1973). The various symbols used in Figure 11 have the following definitions:

curve 1	– follow the line of the shoulder, slightly downward
fingers 1	– fingertips level with the breastbone
fingers 2	– breadth of the forehead apart
hands not 2nd position	– downward according to line required
curve 2	– above the ears and by lifting the eyes he can see the insides of his hands
fingers 3	– fingertips over and just in front of crown of head
fingers 4	– width of his forehead apart
hand 2nd position	– facing directly downward
shoulders	– pulled outward and pressed downward
chest	– fully expanded with easy breathing
handshake	– natural position
too stiff	– correct by softening

There are two distinct levels at which one can detect analogies within a genetic graph. The simplest level involves the individual nodes of the genetic graph. Detection of a strong analogy between two nodes in the genetic graph indicates that an *analogy link* should exist between these two nodes. The second level of representation is significantly more difficult to process as it involves the structure of the genetic graph itself. The objective is to find analogous subgraphs contained within the genetic graph.

It is first necessary to determine a computational representation for the genetic graph. We employ a common graph representation scheme, an *adjacency-vector / linked-list structure*. A *vertex*, that represents a node in the genetic graph, has a 'label', a pointer to the knowledge that it represents, and a pointer to a list of *edges*. An *edge* represents a link in the genetic graph. Each *edge* has a 'type' (*analogy, refinement* etc.), a pointer to an adjacent *vertex* (that the link is directed to), and a pointer to another *edge* structure (to form a linked-list of *edges*). The *adjacency vector* is a list that contains pointers to all of the vertices. There is only one physical structure created for each *vertex* , both the *adjacency vector* and any incoming *edge* have pointers to it. By indexing the adjacency vector, vertices can be directly accessed. To access the adjacent *vertices* of a *vertex* requires following the edge-linked-list for that *vertex* .

Secondly, we must determine an appropriate representation scheme for the knowledge represented at each node of the genetic graph. As the genetic graph structure does not put any restrictions on the representation of the knowledge contained at individual nodes, one may select the form of representation (frame, relational logic, network, etc.) which is best suited to the application domain at hand. For our purposes here, we represent the information at each node of the genetic graph as a forest (a collection of trees). This allows distinct, but related facts and/or procedures, to be represented at each node if desired. To illustrate this, consider the node that is to represent the location and characteristics of the arms when held in first position. In this case, the following facts are represented by a single tree:

First position, Arms -

- Both arms are forward of the body wall
- Both arms are parallel to the floor
- Both elbows are rounded moderately
- Both wrists are bend in
- The hands are separated by six inches
- The palms are open toward the face

that has the syntactic representation:

```
node (1st_position_arms)
[ 1st Position - Arms ]
{
    arms
    (
        direction(forward), parallel(floor),
        wrists (bent(in)),
        elbows (rounded(moderately)),
        hands
        (
            separation(6inches),
            palms (parallel(face))
        )
    )
}
```

5.2. Extension of the Algorithm

The significant role analogy plays in learning and instruction makes it necessary to include analogy within a student model, as seen with the genetic graph. Constructing a genetic graph for anything but exceedingly simple-minded domains will be tedious and error-prone, if done by hand, due to size and complexity. One step in automating the construction of a genetic graph would be by automated detection and inclusion of the *analogy links*.

Figure 12 represents the extension applied to the original matching algorithm (in Figure 2). The new *cost* function differs from the original in that it no longer requires that subtree branches have a particular ordering and it handles node matching between nodes of differing degrees. This modification does increase the algorithm's complexity but it remains polynomial.

$$
\begin{aligned}
cost\,(t_1,t_2) \equiv\ & \textbf{if } root\,(t_1) = root\,(t_2) = \textbf{nil then } 0 \\
& \textbf{else if } root\,(t_1) = \textbf{nil or } root\,(t_2) = \textbf{nil or } \{leaf\,(t_1) \oplus leaf\,(t_2)\} \textbf{ then} \\
& \quad \text{LARGE-CONSTANT} \times (\,|\,t_1\,| + |\,t_2\,|\,) \\
& \textbf{else} \quad \sigma(root\,(t_1), root\,(t_2)) + \sum_{i=1}^{\delta(root\,(t_1))} \underset{j=i}{\text{MIN}}^{\delta(root\,(t_2))} \left\{ cost\,(child\,(i\,,t_1), child\,(j\,,t_2)) \right\} \\
& \quad + \text{LARGE-CONSTANT} \times \sum_{j=\delta(root\,(t_1))+1}^{\delta(root\,(t_2))} |\,child\,(j\,,t_2)\,| \ \textbf{endif.}
\end{aligned}
$$

Figure 12
Modification to Matching algorithm

5.3. Testing

Applying the algorithm to the problem of detecting analogies for the automated construction of genetic graphs is a major task. The testing that has been carried-out to date for this particular application has been limited to analogy detection within a subset of knowledge associated with introductory ballet instruction. We describe here one particular test case; for a discussion of other test cases see (Wellsch 1985).

There are six basic body stances or positions that are taught to introductory ballet students (see Figure 13). The *bras bas* is a basic arm position; the arm motion to first position consists of moving the arms upward from *bras bas*. The algorithm was first applied to the overall physical characteristics shown in Figure 13, arm and leg positions are combined. Table 1A and 1B show the results. All of the possible pairings resulted in a worst-case of LARGE-CONSTANT so the *cost* values are the only way of differentiating between them. There are two points to be made about the numbers that appear in Table 1A. First, they represent the accumulated ratings for each node pairing (or individual node when no pairing was possible). Thus they are a function of the number of nodes used in the trees.

The second point is the basis by which the numbers are computed. The constant LARGE-CONSTANT was assigned a value of 100. The remaining values that compose the *cost* are the rated *similarity* for each node pairing. Conveniently enough, one can distinguish between the node similarity ratings and the number of occurrences of LARGE-CONSTANT in the raw cost values in Table 1A. For example, the matching between *first position* and *second position* has a raw cost of 703. This suggests that there were 7 nodes that could not be matched, and those that did match had a total difference of 3.

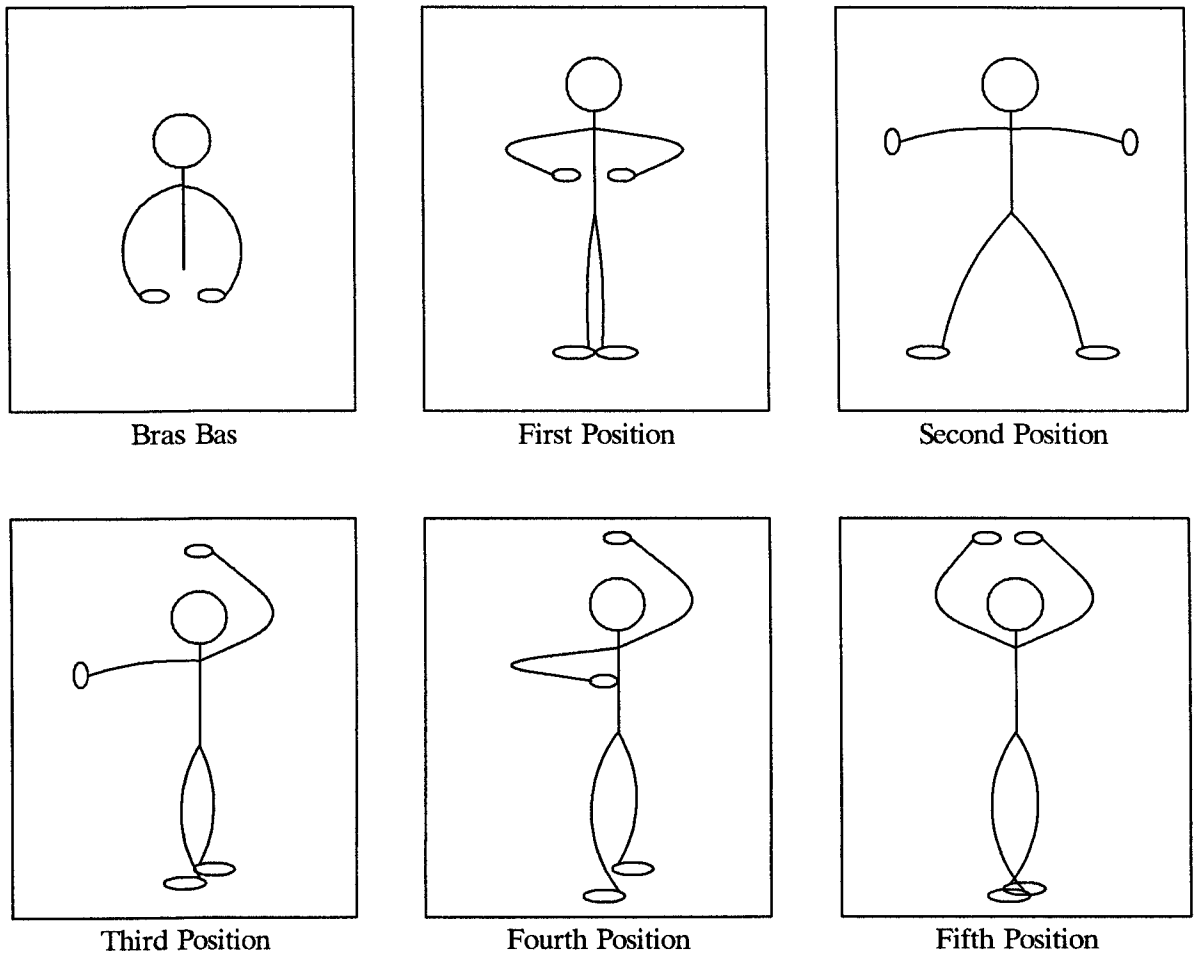

Figure 13
Six basic body stances for ballet

Table 1A
Overall cost values (raw)

cost values	Bras Bas	1st position	2nd position	3rd position	4th position	5th position
Bras Bas	0	200	903	1508	1508	400
1st pos.	200	0	703	2818	2517	1108
2nd pos.	903	703	0	2615	2825	1110
3rd pos.	1508	2818	2615	0	706	2150
4th pos.	1508	2517	2825	706	0	2031
5th pos.	400	1108	1110	2150	2031	0

Table 1B
Overall cost values (average)

cost values	Bras Bas	1st position	2nd position	3rd position	4th position	5th position
Bras Bas	0	4	18	24	25	9
1st pos.	4	0	11	36	33	18
2nd pos.	18	11	0	33	36	18
3rd pos.	24	36	33	0	7	29
4th pos.	25	33	36	7	0	27
5th pos.	9	18	18	29	27	0

Table 2

Interesting Pairings			
Position	Average Cost	Value	Position
Bras Bas	5	100	1st position arms
Bras Bas	12	100	5th position arms
1st position arms	18	100	2nd position arms
1st position arms	18	100	5th position arms
1st position legs	0	1	2nd position legs
2nd position arms	17	100	5th position arms
3rd position arms	11	100	4th position arms
3rd position arms	43	2	5th position arms
3rd position legs	0	1	4th position legs
3rd position legs	3	100	5th position legs
4th position arms	41	2	5th position arms
4th position legs	3	100	5th position legs

The values in Table 1B represent the average computed by dividing the raw score (from Table 1A) by the number of nodes involved. If the maximum value (that a node can be rated at) is one hundred, the average values in Table 1B can range from zero to one hundred inclusive. Because this average is independent of the number of nodes involved, it can then be used to compare the different pairings.

The results in Table 1(A&B) are in keeping with intuition. *Bras bas* is similar to *first position*. The fact that *bras bas* is a position variation on the arm position of *first position* suggests that this a reasonable result. *First position* is similar to *second position* and to a lesser extent, *fifth position*; the arms in these three are symmetric. The leg position of *fifth position* accounts for the decrease in similarity with both first and second. *Third position* appears to correspond best with *fourth position*; they both share an asymmetry in the arm positions. Hence the results of the algorithm agree with an intuitive view of the six stances.

A further refinement was tested that separates the arm and leg positions. Tables 3A, 3B and 4 contain the results of all possible pairings. Again, the results are intuitively acceptable. Table 3A and 3B contain the total *cost* values for the detailed matching. The values have the same basis as those in Table 1A and 1B. The values in Table 4 are the *worst-case values* for the matching. The '*' entries represent a worst-case occurrence of LARGE-CONSTANT. Since one cannot assign a value worse than this, the *cost* value is the only way to distinguish between these cases. It is the matches that have a worst-case less than LARGE-CONSTANT for which this value has differentiating capability. (Note that an exact match has a worst-case value and cost of zero.) When the *worst-case value* is less than LARGE-CONSTANT, there is an isomorphic matching of the two tree structures and the node symbols differ by at worst, that

value.

Looking at Tables 3A, 3B and 4, there are twelve interesting matchings. They stand out from the other matchings, either because they have a low cost relative to the other matchings, or that they have a worst-case value that is small (i.e. < LARGE-CONSTANT). These matchings are summarized in Table 2. In this example, costs below 20 were considered interesting. Whether an absolute value (such as 20) exists for most or all cases or must be determined for each domain, is an open problem.

Again *bras bas* matches well with both first and fifth positions. *First position - arms* matches well with second and fifth position arms, but it is the *first position - legs* match with *second position - legs* that is interesting. *Third position - legs* and *fourth position - legs* matching also shares this strong similarity. The effect is due to the fact that each pair differs only in the separation of the feet. Otherwise, they are the same.

Table 3A
Detailed Cost values (raw)

cost values		Bras Bas	1st arms	1st legs	2nd arms	2nd legs	3rd arms	3rd legs	4th arms	4th legs	5th arms	5th legs
Bras Bas		0	200	1504	903	1504	2413	1508	2209	1508	400	1508
1st	arms	200	0	1704	702	1704	2310	1708	2009	1708	600	1708
	legs	1504	1704	0	1509	1	2877	508	2677	508	1407	508
2nd	arms	903	702	1509	0	1509	2107	1609	2317	1609	602	1609
	legs	1504	1704	1	1509	0	2877	508	2677	508	1407	508
3rd	arms	2413	2310	2877	2107	2877	0	2677	705	2677	2049	2772
	legs	1508	1708	508	1609	508	2677	0	2577	1	1606	101
4th	arms	2209	2009	2677	2317	2677	705	2577	0	2577	1929	2672
	legs	1508	1708	508	1609	508	2677	1	2577	0	1606	102
5th	arms	400	600	1407	602	1407	2049	1606	1929	1606	0	1606
	legs	1508	1708	508	1609	508	2772	101	2672	102	1606	0

Table 3B
Detailed Cost values (average)

cost values		Bras Bas	1st arms	1st legs	2nd arms	2nd legs	3rd arms	3rd legs	4th arms	4th legs	5th arms	5th legs
Bras Bas		0	5	51	25	51	50	50	47	50	12	50
1st	arms	5	0	54	18	54	46	53	41	53	18	53
	legs	51	54	0	45	0	63	18	60	18	50	18
2nd	arms	25	18	45	0	45	40	47	45	47	17	47
	legs	51	54	0	45	0	63	18	60	18	50	18
3rd	arms	50	46	63	40	63	0	58	11	58	43	60
	legs	50	53	18	47	18	58	0	57	0	55	3
4th	arms	47	41	60	45	60	11	57	0	57	41	59
	legs	50	53	18	47	18	58	0	57	0	55	3
5th	arms	12	18	50	17	50	43	55	41	55	0	55
	legs	50	53	18	47	18	60	3	59	3	55	0

Both *third position - legs* and *fourth position - legs* share a similarity with *fifth position - legs*. This similarity is weaker than the two previously described leg matchings because they share the more general notion of one foot in front of the other. An interesting matching exists between *third position - arms* and *fifth position - arms*, and *fourth position - arms* and *fifth position - arms*. The matching costs are both very high, yet their worst-case values are very low. This would seem to be an odd situation but, in fact, does have a simple explanation. Consider Figure 13; both third and fourth position have one arm the same as in fifth position. This leads to a strong matching. It is the lack of a match for the other arm in both third and fourth positions that results in such a high cost.

Table 4
Worst-case values

worst-case values			1st		2nd		3rd		4th		5th	
		Bras Bas	arms	legs	arms	legs	arms	legs	arms	legs	arms	legs
Bras Bas		0	*	*	*	*	*	*	*	*	*	*
1st	arms	*	0	*	*	*	*	*	*	*	*	*
	legs	*	*	0	*	1	*	*	*	*	*	*
2nd	arms	*	*	*	0	*	*	*	*	*	*	*
	legs	*	*	1	*	0	*	*	*	*	*	*
3rd	arms	*	*	*	*	*	0	*	*	*	2	*
	legs	*	*	*	*	*	*	0	*	1	*	*
4th	arms	*	*	*	*	*	*	*	0	*	2	*
	legs	*	*	*	*	*	*	1	*	0	*	*
5th	arms	*	*	*	*	*	2	*	2	*	0	*
	legs	*	*	*	*	*	*	*	*	*	*	0

* indicates that multiplicity or inconsistency occurred

6. Concluding Remarks

As illustrated by the test cases presented in the previous section, the results do correspond with one's intuition i.e. the analogies detected by the algorithm are easily explained. This is also true of the numerous test cases we employed within the domain of two-dimensional geometric shapes. At this point the full power of the algorithm has not been investigated; the analogies we have investigated in the context of instructional systems are as yet restrictive in nature. One would like to be able to detect analogous learning situations (including concepts, techniques, instructional methodologies, examples), so that this knowledge can then be employed to improve the diagnostic and instructional capabilities of a system. We have not yet applied our algorithm to such extensive examples, but then again such extensive student models have not yet been developed for *any* domain.

One of the potential advantages of an effective algorithm for detecting analogies is in facilitating the development of dynamic student models. Generating a student model (regardless of the chosen representation scheme) can be a slow and tedious task. Automating the creation of a student model (or parts thereof) given some domain knowledge is a worthwhile goal. Moreover, one does not want a static student model; after all large chunks of it may be unnecessary for a particular student. Rather, student models should be dynamic. (This is especially true when working with a variety of student populations or special populations such as learning disabled students where one expects more inter-student variation). The development of dynamic student models is a major open research issue within the area of ICAI. Creating and maintaining the genetic graph by an automated process is one important aspect of a dynamic student model that is then formed by an overlay on the genetic graph. One of the tasks necessary for such automation is the determination of the *analogy links* within the genetic graph.

During the course of this investigation, many interesting issues, both in regard to analogy detection and student modelling, have risen. For example, how much information should be represented at a genetic graph node? ICAI research has not yet addressed this question, either in the context of the genetic graph or any other student model. Because the algorithm is not solely based on a node-to-node matching, altering the amount of knowledge stored at a node should not effect the performance of the algorithm, but we have not verified this.

With regard to more general issues concerning analogy testing, the handling of *inconsistency* and *multiplicity* warrant further investigation. Again our initial results within both the domains of two-dimensional geometric shapes and genetic graphs, are very encouraging; the current approach (although simple) appears to be very effective. Although a more sophisticated method of handling these issues *may* improve applicability, one does not want to increase the computational complexity of the algorithms (which is polynomial in the size of the trees).

A variety of questions remain unanswered (i.e. open problems). One such question asks: "does there exist **one** algorithm for all analogies?" The likelihood of this remains remote; for one reason, the rather open definition of analogy. As seen in section 2, researchers view analogy in a variety of ways. Much like a definition for 'intelligence' or 'learning', a clear definition of 'analogy' remains to be firmly established.

As suggested in section 3.1, one drawback to a hierarchical knowledge representation is the problem of multiple representations. If it is possible to represent a scene in several ways, which one should we choose? Multiple representations are merely the implicit manifestation of the notion of importance and can provide useful knowledge in this way. Gestalt psychology shows us that although a particular representation of a scene may be formed, it may be necessary to re-represent the scene should it fail to provide a matching.

The matching process has a variety of features to use as selection criteria (i.e. *cost, worst-case* etc.). How important are each of these and how many are necessary in order to obtain an acceptable domain-independent matching? For the algorithm given in Figure 2, matching cost, and worst-case were used, with worst-case having a greater priority.

In addition to the various avenues for future research already mentioned, a heuristic rather than deterministic approach is worth investigating. As suggested for scene analysis (§ 3.1), some use of heuristics may be essential. Current research is already looking at heuristics for improving mapping constraints (Greiner 1985).

7. References

Aho, A. V., Hopcroft, J. E. and Ullman, J. D. (1974), *The Design and Analysis of Computer Algorithms,* Addison-Wesley.

Bourne, L. E., Domonowski, R. L. and Loftus, E. F. (1979), *Cognitive Processes,* Prentice-Hall.

Burstein, M. H. (1983), "A Model of Learning by Incremental Analogical Reasoning and Debugging", *Proceedings of AAAI-83,* 45-48.

Carbonell, J. G. (1981), "A Computational Model of Analogical Problem Solving", *Proceedings of IJCAI-81,* 147-152.

Carbonell, J. G. (1983), "Derivational Analogy and its Role in Problem Solving", *Proceedings of AAAI-83,* 64-69.

Carbonell, J. G. (1983), "Learning by Analogy: Formulating and Generalizing Plans from Past Experience", in *Machine Learning - An Artificial Intelligence Approach,* 137-161.

Carbonell, J. G. and Minton, S. (1983), "Metaphor and Common-Sense Reasoning", CMU Technical Report CS-83-110.

Dember, W. N. and Warm, J. S. (1979), *Psychology of Perception,* Holt, Rinehart and Winston.

Dreistadt, R. (1968), "An Analysis of the Use of Analogies and Metaphors in Science", *The Journal of Psychology,* 68, 97-116.

Dundas, M. and Stockdale, G. (1985), "A Student Model for Long Division using the Genetic Graph Approach", CS486 course project, unpublished manuscript.

Evans, T. G. (1968), "A Program for the Solution of a Class of Geometric-Analogy Intelligence-Test Questions", in *Semantic Information Processing,* 271-353, MIT Press.

Fu, K. Sun (1982), *Syntactic Pattern Recognition and Applications,* Prentice-Hall.

Garey, M. R. and Johnson, D. S. (1979), *Computers and Intractability,* W. H. Freeman and Company.

Gentner, D. (1982), "A Structure-Mapping Approach to Analogy and Metaphor", *IEEE Proceedings, International Conference on Cybernetics and Society,* 75-79.

Gentner, D. (1983), "Structure Mapping: A Theoretical Framework for Analogy", *Cognitive Science,* 7, 155-170.

Gentner, D. and Gentner, D. R. (1983), "Flowing Waters or Teeming Crowds: Mental Models of Electricity", in *Mental Models,* 99-129, L. Erlbaum Assoc. Inc.

Gick, M. L. and Holyoak, K. J. (1980), "Analogical Problem Solving", *Cognitive Psychology,* 12, 306-355.

Gick, M. L. and Holyoak, K. J. (1983), "Schema Induction and Analogical Transfer", *Cognitive Psychology,* 15, 1-38.

Goldstein, I. P. (1982), "The Genetic Graph: A Representation for the Evolution of Procedural Knowledge", in D. Sleeman and J. S. Brown (eds), *Intelligent Tutoring Systems,* Academic Press, 51-77.

Greiner, R. (1985), "Learning by Understanding Analogies", Ph.D. Thesis, Department of Computer Science, Stanford University.

Halasz, F. and Moran, T. P. (1982), "Analogy Considered Harmful", *Proceedings, Human Factors in Computer Systems,* 383-386.

Hesse, M. B. (1966), *Models and Analogies in Science,* Univ. of Notre Dame Press.

Indurkhya, B. (1985), "Approximate Semantic Transference: A Computational Theory of Metaphors and Analogies", Boston University Technical Report #85/012.

Kedar-Cabelli, S. T. (1985), "Analogy — From a Unified Perspective", Rutgers University Report ML-TR-3.

Kling, R. E. (1971), "A Paradigm for Reasoning by Analogy", *Artificial Intelligence,* 2, 147-178.

Langley, P., Bradshaw, G. L. and Simon, H. A. (1981), "BACON.5: The Discovery of Conservation Laws", *Proceedings of IJCAI-81,* 121-126.

Lawson, J. (1973), *The Teaching of Classical Ballet: Common Faults in Young Dancers and Their Training,* A&C Black Limited.

Lenat, D. B. (1982), "The Nature of Heuristics (I)", *Artificial Intelligence,* 19, 189-249.

Miller, G. A. (1956), "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capability for Processing Information", *American Psychologist,* 17, 748-762.

Moore, J. and Newell, A. (1974), "How Can Merlin Understand?", in *Knowledge and Cognition,* L. Erlbaum Assoc. Inc., 201-252.

Mulholland, T. M., Pellegrino, J. W. and Glaser, R. (1980), "Components of Geometric Analogy Solution", *Cognitive Psychology,* 12, 252-284.

Oppenheimer, R. (1956), "Analogy in Science", *The American Psychologist,* 11, 127-135.

Ortony, A. (1979), "Beyond Literal Similarity", *Psychological Review,* 86, 161-180.

Polya, G. (1954), *Induction and Analogy in Mathematics,* Princeton University Press.

Quine, W. V. and Ullian, J. S. (1978), *The Web of Belief,* Random House, Inc.

Reed, S. K., Ernst, G. W. and Banerji, R. (1974), "The Role of Analogy in Transfer Between Similar Problem States", *Cognitive Psychology 6,* 436-450.

Reynolds, A. G. and Flagg, P. W. (1977), *Cognitive Psychology,* Winthrop.

Ross, B. H. (1984), "Reminding and Their Affects in Learning a Cognitive Skill", *Cognitive Psychology,* 16, 371-416.

Rumelhart, D. E. and Abrahamson, A. A. (1973), "A Model of Analogical Reasoning", *Cognitive Psychology,* 5, 1-28.

Rumelhart, D. E. and Norman, D. A. (1981), "Analogical Processes in Learning", in *Cognitive Skills and Their Acquisition,* L. Erlbaum Assoc. Inc., 335-359.

Schustack, M. W. and Anderson, J. R. (1979), "Effects of Analogy on Prior Knowledge on Memory for New Information", *Journal of Verbal Learning and Verbal Behavior,* 18, 565-583.

Sternberg, R. J. (1977), "Component Processes in Analogical Reasoning", *Psychological Review,* 84, 353-378.

Tourangeau, R. and Sternberg, R. J. (1981), "Aptness in Metaphor", *Cognitive Psychology,* 13, 27-55.

Tversky, A. (1977), "Features of Similarity", *Psychological Review,* 84, 327-352.

Wasson, B. J. (1985), The Development of Student Models: The Genetic Graph Approach, M. Math Thesis, Department of Computer Science, University of Waterloo; also available as Research Report CS-85-10, 102p.

Wellsch, K. C. (1985), *A Computational Model for Reasoning by Analogy,* M. Math Thesis, Department of Computer Science, University of Waterloo, 167p.

Wellsch, K. C. and Jones, M. L. (1986a), "Detecting Analogous Learning", *Sixth Canadian Conference on Artificial Intelligence,* 17-23.

Wellsch, K. C. and Jones, M. L. (1986b), "Computational Analogy", *European Conference on Artificial Intelligence,* 153-162.

Winston, P. H. (1980), "Learning and Reasoning by Analogy", *Communications of the ACM,* 23, 689-703.