Visual Case-Based Reasoning II: Transfer and Adaptation[1]

Jim Davies and Ashok K. Goel

jim@jimdavies.org, goel@cc.gatech.edu
Artificial Intelligence Laboratory
College of Computing, Georgia Institute of Technology
Atlanta, Georgia 30332, USA

Abstract. Case-based problem solving refers to reasoning about new problems by reusing past cases. Visual case-based problem solving pertains to reuse of past cases that contain only visual knowledge. In this paper, we explore the problem of automated adaptation of diagrammatic cases, i.e., automatic transfer of diagrammatic knowledge from a source case to a target problem. We describe Galatea, a computer program that adapts diagrammatic cases. Galatea explicitly represents knowledge states in the source case in the form of propositions, and transfers visual transformations from the source case to the target problem. A companion paper in the same conference [15] addresses the task of retrieving diagrammatic cases.

1. Introduction

Case-based problem solving refers to reasoning about new problems by reusing past cases. Research on case-based reasoning generally has focused on cases that contain functional and causal knowledge, e.g., goals and plans. However, psychological evidence clearly indicates that visual knowledge too plays an important role in complex human information processing [e.g. 10]. In design, for example, drawings and diagrams play a critical role in expressing, communicating, using and learning cases. There is also documentary evidence for visual reasoning in scientific problem solving [11]. Further, psychological evidence also indicates that case-based problem solving itself is facilitated by diagrams and animations [13], and visually evocative phrases in stimuli [7]. Therefore, an important issue in case-based problem solving is how to use cases that contain visual knowledge. In this paper, we focus on visual case-based reasoning, i.e., reasoning with cases that contain only visual knowledge, where visual knowledge consists (only) of information about how an entity appears. Although visual cases in general may contain knowledge of many kinds, such as photographs, drawings, diagrams, animations and videos, this work deals only with diagrammatic knowledge represented symbolically. Specifically, this work deals with diagrammatic knowledge of shapes, their sizes, locations and motions, and the spatial relationships among the shapes.

---

[1] Davies, J. & Goel, A. K. (2003). Visual case-based reasoning I: Transfer and adaptation. *Proceedings of The 1st Indian International Conference on Artificial Intelligence.* Springer.

ANALOGY [3] and Letter Spirit [9] are earlier AI programs that reasoned about visual cases. Unlike these systems, we are interested in general problem solving where the source cases may contain problem-solving procedures, i.e., ordered sequences of problem-solving steps. The core question in our work is how to represent visual problem solving and how to transfer the problem-solving procedures from the source case to the target problem.

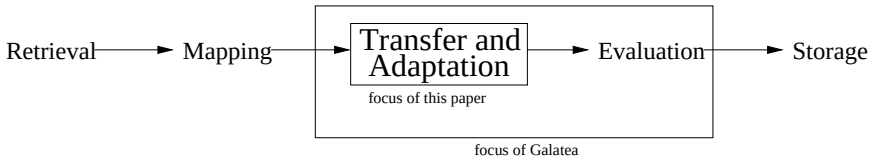

Fig. 1. The steps of analogy. Galatea implements transfer, adaptation, and evaluation. This paper focuses on transfer and adaptation.

In general, case-based analogical problem solving spawns several tasks as illustrated in Figure 1. This paper focuses on the task of transfer and adaptation. This task takes as input a target problem, an appropriate source case, and an initial mapping between various elements in the target and source cases. A companion paper in the same conference [15] addresses the task of retrieving diagrammatic cases from a computer-based library. We are presently integrating the computer programs described in the two papers.

2 An Illustrative Example

Our model is implemented in a computer program called Galatea. We begin with a simple example that is easy to explain and yet illustrates some of the issues. Galatea is given the target problem of feeding six people with one pizza. It is also given a source case in which a cake was cut into four pieces for four people. The issue is how might a case-based problem solver represent its diagrammatic knowledge of the source case and target problem, and how might it transfer the relevant problem-solving steps from the source to the target?

Galatea represents a source case as a series of knowledge states starting from the initial knowledge state and ending in the final or goal knowledge state. The simple cake case contains only two knowledge states, the initial and the goal state. A knowledge state is represented diagrammatically in the form of shapes, their locations, sizes, and motions (if any), and the spatial relationships among the shapes. For the cake case, Galatea represents cake in the initial knowledge state as a rectangle of a specific size, and the goal knowledge state as containing four, smaller rectangles. For the target pizza problem, Galatea initially knows only the initial knowledge state and represents the Sicilian slice pizza as a rectangle.

Succeeding states in the series of knowledge states are related through visual transformations such as move, rotate, scale and decompose. Each transformation relates two knowledge states. In the cake case, the *decompose* transformation takes one shape (the rectangle) and a number *n* (an argument of the transformation) and

results in *n* smaller pieces of the same shape.[2] Galatea initially does not know of any transformations for the target pizza problem.

The question now becomes how Galatea can transfer knowledge of the decompose transformation from the source cake case to the target pizza problem. Galatea attempts to transfer the knowledge states and visual transformations following the initial state in the source cake case to the target pizza problem. One source of difficulty here is that while the number of people in the cake case is six, the number of people in the pizza problem is four. Galatea uses the concepts of sets and members to address this issue. It explicitly represents the people as belonging to a set in the source and target cases. The *decompose* transformation takes as an argument a set, the members of which is counted for each case: in the cake case, this set contains *six* members; in the target problem it contains *four*. When the *decompose* transformation is transferred to the target pizza problem, it is instantiated with the count of the set of people in the target, not the source. When the transformation is executed, it generates six smaller rectangles in the final knowledge state of the target problem.

3 Knowledge Representation and Algorithms

In this section, we will describe Galatea's representation language and algorithm using Gick and Holyoak's fortress/tumor problem [7] as a running example: the problem is based on experiments done in which participants are given a story about a general who needs to break his army up into smaller groups so they can converge on a fortress through different roads, because each road has a mine on it that will go off if the whole army crosses it. Participants are expected to transfer this solution to a target problem in which a tumor needs to be radiated without hurting the healthy tissue on its way in. The solution using the source case is to use several weaker rays of radiation coming in from different directions.

3.1 Knowledge Representation

Galatea describes visual cases using Covlan (Cognitive Visual Language), which consists of knowledge states, primitive elements, primitive relations, primitive transformations, general visual concepts, and correspondence and transform representations. In Covlan, all knowledge is represented as propositions.

Knowledge States: Knowledge States in Covlan are symbolic images, or s-images, which contain visual elements, general visual concepts, and relations between them. Cases are represented by a series of s-images, connected with transformations.

Visual Transformations. An S-image in the sequence is connected to other s-images before and after it with transformations. Transformations, like ordinary functions, take arguments to specify their behavior.

These transformations control normal graphics transformations such as translation (*move-to-location*, *move-to-touch*, *move-above*, *move-to-right-of*, *move-to-left-*

[2] In the current version of Galatea, the *decompose* transformation cannot break up one shape into *n* smaller *different* shapes, as one might cut a round pizza into roughly triangular shapes. To do this would require changing the transformation so that it either had a complex notion of how shapes can be sectioned, or took as an argument the resultant shapes.

of, move-below), rotation (*rotate*), and scaling (*set-size*). In addition there are transformations for adding and removing elements from the s-image (*add-element, remove-element*). Certain transformations (*start-rotating, stop-rotating, start-translation, stop-translation*) are changes to the dynamic behavior of the system under simulation. For example, *rotate* changes the initial orientation of an element, but in contrast *start-rotating* sets an element in motion.

The fortress/tumor example requires two transformations. The first is to *decompose* the army into several groups, and the second is to use *move-to-location* to move them to the separate roads (see Figure 2.)

Primitive Elements. These are the visual objects in a diagram. The element types are *polygon, rectangle, triangle, ellipse, circle, arrow, line, point, curve,* and *text*. Each element is represented as a frame with attribute slots, such as *location, size, orientation,* or *thickness.* The primitive elements are organized into a two-tiered hierarchy. For example, rectangle and triangle are subclasses of polygon; circle is a subclass of ellipse.

General Visual Concepts. These act as slots for the primitive elements as well as arguments for the visual transformations. The concepts are *location, size, thickness, speed, direction, length, distance, angle,* and *direction.* Each concept has several values it can take. For example, the *size* can be *small, medium,* or *large,* and *thickness* can be *thin, thick* or *very-thick. Location* specifies an absolute qualitative location in an s-image (*bottom, top, center,* etc.)

Primitive Visual Relations. This class of symbols describes how certain visual elements relate to each other and to the values taken by general visual concepts. The visual relations are *touching, above-below, right-of-left-of, in-front-of-behind,* and *off-s-image*. The motion relations are *translation* and *rotation*.

Correspondence and Transform Representations. The knowledge of which objects in one knowledge state correspond to which objects in another is a *mapping,* which consists of a set of alignments between objects. Different sets of alignments compose different mappings. The i^{th} s-image in the source and the i^{th} s-image in the target have a *correspondence* between them; each correspondence can have any number of *mappings* associated with it (determining which mapping is the best is the "mapping problem.") The correspondence and mapping between the initial s-images (i=1) in the source and target is given as part of the input to Galatea; the system generates the subsequent correspondences and mappings.

Similarly, successive s-images in a series have *transform-connections*. These are needed so that Galatea can track how visual elements in a previous knowledge state change in the next.

3.2 Algorithm

Following is the control structure for Galatea's transfer of problem-solving procedures from a source case to the target problem. We will describe the transfer of the first transformation as a running example. Figure 2 shows the s-image structure for the fortress/tumor problem. The figure references in the algorithm description below refer to Figure 2.

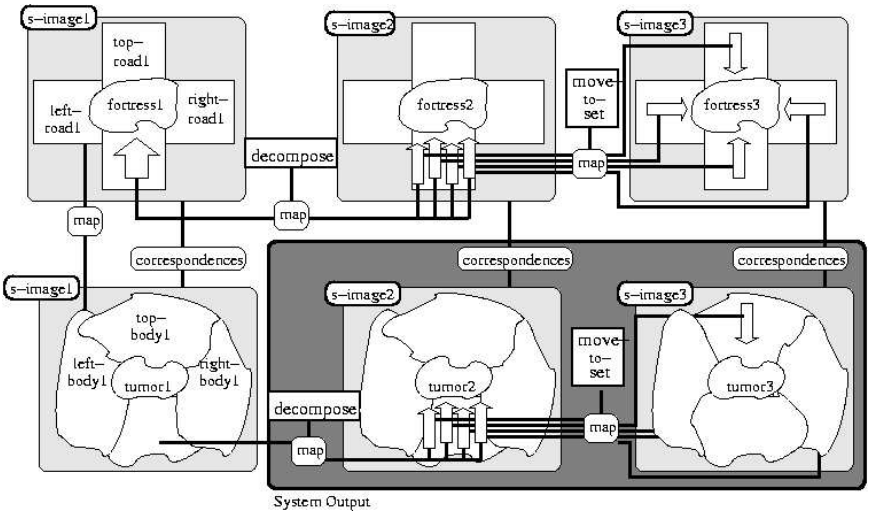

Fig. 2. The top three boxes, going left to right, are the three s-images in the fortress case. They are connected by transformations: the first and second by *decompose* and the second and third by *move-to-set*. *Decompose* takes the soldier path and breaks it up into *n* thinner soldier paths, where *n*, the argument to *Decompose*, can take an integer value, e.g., 4. Likewise, in the tumor problem, whose s-images run along the bottom of the Figure, *decompose* turns the single ray into *n* thinner rays. The *move-to-set* transformation puts these thin lines in the appropriate places. The gray area shows the output of Galatea

1. **Identify the first s-images of the target and source cases.**

2. **Identify the transformations and associated arguments in the current s-image of the source case**. This step finds out how the source case gets from the current s-image to the next s-image. In our example, the transformation is *decompose*, with ``four'' as the *number-of-resultants* argument (not shown).

3. **Identify the objects of the transformations.** The object of the transformation is what object the transformation acts upon. For the *decompose* transformation is the *soldier-path1* (the thick arrow in the top left s-image in Figure 2.)

4. **Identify the corresponding objects in the target problem**. The *ray1* (the thick arrow in the bottom left s-image) is the corresponding component of the source case's *soldier-path1*, as specified by the correspondences between the s-images (not shown). Adaptation of the arguments can happen three ways: If the argument is a component of the source s-image, then its analog is found. If the argument is a function, then the function is run (note that the function itself may have arguments which follow the same adaptation rules as transformation arguments). Else the arguments are transferred literally.

5. **Apply the transformation with the arguments to the target problem component**. A new s-image is generated for the target problem (bottom middle) to record

the effects of the transformation. The *decompose* transformation is applied to the *ray1*, with the argument ``four.'' The result can be seen in the bottom middle s-image the Figure. The new rays are created for this s-image. As the soldier paths are put into a set in the source case, so the rays are put into a set in the target.

6. **Map the original objects to the new objects in the target case**. A transform-connection and mapping are created between the target problem s-image and the new s-image (not shown). Maps are created between the corresponding objects. In this example it would mean a map between *ray1* in the left bottom s-image and the four rays in the second bottom s-image. This system does not solve the mapping problem, but a mapping from the correspondences of the first s-image enable the mappings for the subsequent s-images to be automatically generated. The transformation is associated with the map so the target case itself can be put in the case library as a possible source in the future.

7. **Map the new objects of the target case to the corresponding objects in the source case**. Here the rays of the second target s-image are mapped to soldier paths in the second source s-image. This step is necessary for the later iterations (i.e. going on to another transformation and s-image). Otherwise the reasoner would have no way of knowing which parts of the target s-image the later transformations would operate on.

8. **Check to see if goal conditions are satisfied**. If they are, exit, and the problem is solved. If not, and there are further s-images in the source case, set the current s-image equal to the next s-image and go to step 1. If there are no further s-images, then exit and fail.

4 Discussion: Related Work

Below we describe the relationship of this work to both case-based reasoning and analogical reasoning. However, since this paper focuses on transfer and adaptation, we limit the discussion to these tasks; the companion paper [15] describes the relationship to other work on case retrieval.

Case-Based Reasoning: Many case-based systems contain multi-modal cases, i.e., cases that contain both visual (e.g., photographs, drawings, diagrams, animations and videos) and non-visual knowledge (e.g., goals, constraints, plans and lessons). However, many of these projects (e.g., ARCHIE, [12], AskJef, [1]) leave the adaptation task to the user and do not automate the transfer of diagrammatic knowledge form a source case to a target problem.

FABEL [6] is an example of a case-based system that adapts diagrammatic cases in the domain of architectural design. The diagram in FABEL specifies the spatial layout of a building or similar structure. It adapts source diagrams by extracting and transferring specific structural patterns to the target problem. It uses domain-specific heuristics to guide pattern extraction and transfer. Galatea too adapts diagrams by extracting and transferring patterns. Pattern transfer in Galatea is facilitated by three main elements. Firstly, Galatea explicitly represents the knowledge states of its source cases in the form of s-images. Secondly, each s-image is composed of primitive visual elements and relations. Thirdly, succeeding knowledge states in Galatea's

source cases are related by primitive visual transformations. In this way, Galatea captures the diagrammatic *problem solving* of the source cases. Given a mapping between the visual elements in the target problem and a source case, this knowledge enables Galatea to extract and transfer the appropriate series of visual transformations from the source case to the target problem. In particular, the knowledge states identify the names and arguments of specific transformations that need to be transferred from the source case to the target problem.

Analogical Reasoning: Theories of analogical reasoning, e.g., such as SME [4] and LISA [8], emphasize transfer of complex relations from the source to the target. Galatea too transfers complex relations. In particular, it addresses the problem of transferring problem-solving procedures that contain an ordered series of operations. Some relation-based theories of analogical reasoning are structure-based while others are content-based. SME, for example, provides a uniform structure-based mechanism for analogical reasoning that is intended to work independently of any specific content account. Content-based theories, such as [15], emphasize the content of the representations, and the mechanisms of analogical reasoning are content-dependent. Galatea too is a content-based theory of analogical reasoning.

The visual primitives that describe a knowledge state in Galatea are similar to that of GeoRep [5]. Galatea however uses them for a task quite different from that of GeoRep: GeoRep extracts and abstracts visual relations in line drawings; in contrast, Galatea transfers a problem-solving procedure from a source case to a target problem. To do so, in addition to the visual primitives for describing a knowledge state, Galatea uses primitive transformations that act on knowledge states. Galatea's notion of sets is also similar to the notion of groups in GeoRep. GeoRep dynamically generates groups for abstracting visual relations from line drawings while Galatea uses sets to enable transfer.

As mentioned in the introduction, our work on Galatea builds on the ANALOGY program [3] and the Letter Spirit program [14, 9]. ANALOGY performed simple geometric analogies of the kind that appear on many intelligence tests. Let us suppose that each of A, B, C, D, E and F is a simple arrangement of simple geometric objects, e.g., a small triangle inside a large triangle, a small circle inside a larger circle, etc. Given an analogy A:B, and given C and multiple choices D, E and F, ANALOGY found which of D, E, and F had a relationship with C analogous to that between A and B. It represented the objects and the spatial relationships between them in the form of semantic networks, which enabled it to compare the spatial structure of the various arrangements.

ANALOGY found similarities and differences between visual cases by inferring the transformations (called *rules*) describing how A transforms into B. It had an ontology of shapes (e.g., triangles and circles) and transformations (*scaling, rotation, reflection, addition*, and *deletion*). Since the mapping between A and B and the mapping between A and C are not given, in cases where there is any ambiguity, all possible mappings were generated. For each mapping between A and B, it inferred the transformations between them A and B. This transformation inference process also happened between C and all answer choices. ANALOGY's choice for the best

answer was based on the similarity of the inferred transformations for the source to the inferred transformations of the targets.

Like ANALOGY, Galatea has an ontology of shapes and transformations. Unlike ANALOGY, Galatea represents sets of objects and actions on them because it needs to adapt transformations during transfer from the source case to the target problem. ANALOGY neither transfers nor adapts transformations. Thus, ANALOGY cannot generate new diagrams; it can merely select among given diagrams. Therefore, while ANALOGY can match two diagrams, it cannot use diagrammatic knowledge for problem solving.

Letter Spirit takes as input some number of "seed letters" in a font and outputs the rest of the alphabet in the same font. There are three main parts: The Examiner, the Adjudicator, and the Drafter. The Examiner is a letter-recognition system that identifies the seed letters and determines what they are and which parts of each letter play which "roles" (e.g. crossbar for a *t* or an *f*.) The Adjudicator determines the "spirit," or style, in which the input letters are written. The Drafter, then, draws the rest of the letters of the alphabet, using the discovered style. Thus the targets are multiple, but always from the Roman alphabet. Generated letters must be recognizable as the correct letter (as determined by the Examiner) and must be in the spirit of the seed letters (as determined by the Adjudicator.) As letters are created, they effectively become additional source letters. The system has a detailed knowledge of letters, and what roles are played in each. For example, the letter "b" has a *left-post* and a *right-bowl*. The spirit consists of things like transformations (e.g. *suppress-crossbar*) and rules (e.g. *no-diagonals*). At a lower level of representation letters are straight-line picture elements (called *quanta*) arranged in a grid (the fonts are actually called gridfonts for this reason).

Letter Spirit has many functions that Galatea does not, for example, a built-in evaluation system and multiple targets. However, some aspects of Letter Spirit's representations and reasoning appear to be domain-specific. Galatea is meant to be applicable to a larger class of problems and as a result its representation and processing are not limited to any specific domain.

The most important difference between Galatea and Letter Spirit is that Galatea can handle transfer of problem-solving procedures that Letter Spirit cannot. The procedures are strongly ordered sequences of visual transformations. For example, in the fortress problem there are two transformations that take place: The army (represented as a thick line) is split into smaller armies (thin lines.) Then these smaller lines are moved to various points on the s-image. What makes this problem fundamentally different from those dealt with by ANALOGY or Letter Spirit is that the order of operations is critical—the thin lines cannot be moved until they exist, and they do not exist before the first transformation (*decompose*) occurs, and generates them. Though both Letter Spirit and ANALOGY can both represent multiple transformations, they cannot transfer transformations that rely on the output of previous transformations.

We note that other systems have addressed different visual aspects of the fortress/tumor problem. Diva, for example, [2] addresses the issue of analogical mapping.

5 Conclusion

Galatea shows that at least for problem solving in diagrammatic cases, purely visual knowledge is sufficient for transfer and adaptation, if a mapping between the target problem and source case is given. Galatea provides one method for analogical transfer under the above conditions. The method explicitly represents the problem solving in the source case in terms of knowledge states (composed of primitive elements) and visual transformations between the states. It addresses the target problem by transferring the primitive transformations in the source case one by one and constructing new knowledge states after each transformation.

Galatea also raises some issues that require additional research. Firstly, Galatea suggests that in order to transfer strongly ordered procedures, the visual case-based reasoner must not only represent the knowledge states in the source case but also generate intermediate knowledge states in the target problem. Going back to the tumor example, to transfer the *move-to-location* transformation from moving the armies to the rays, the rays need to exist as symbols in the agent's knowledge. To accommodate this need, Galatea generates intermediate states following each transformation in the target.

Secondly, Galatea suggests that not only does the intermediate state need to be generated, but a new mapping between the intermediate source state and the intermediate target state must be generated as well. The reason is that while the reasoner may know which objects in the source case are transformed, it cannot always use the mapping between the initial states of the source and the target to determine which part of the target gets transformed because some parts are generated during the problem solving. For example, in the tumor example, the rays were not a part of the initial state of the target. Galatea uses the mappings between the source and target along with the mappings between the successive states in the analogs to infer this mapping, which enables it to infer that it is the rays that need to be moved in the second transformation.

Acknowledgements: This work has benefited from many discussions with Nancy J. Nersessian and Patrick Yaner.

References

1. Barber, J., Jacobson, M., Penberthy, L., Simpson, R., Bhatta, S., Goel, A., Pearce, M., Shankar, M. & Stroulia, E. Integrating artificial intelligence and multimedia technologies for interface design advising. *NCR Journal of Research and Development*, 6(1), 75—85, October 1992.
2. Croft, D., & Thagard, P. (2002). Dynamic imagery: A computational model of motion and visual analogy. In L. Magnani and N. Nersessian (Eds.), *Model-based reasoning: Science, technology, values*. New York: Kluwer/Plenum, 259-274, 2002.
3. Evans, Thomas G., A Heuristic Program to Solve Geometric Analogy Problems, in M. Minsky's (Ed.) *Semantic Information Processing*, MIT Press, Cambridge, MA

(1968). Cited in P. H. Winston, *Artificial Intelligence,* And Third Edition. Addison Wesley, Reading, MA, 1992.

4. Falkenhainer, B., Forbus, K. D., and Gentner, D., The Structure mapping engine: algorithm and examples. *Artificial Intelligence* (41): 1—63, 1990.
5. Ferguson, R., and Forbus, K. D., GeoRep: A flexible tool for spatial representation of line drawings. *Proceedings of the Seventeenth National Conference on Artificial Intelligence*. AAAI Press/MIT Press. Cambridge, MA. 510—516, 2000.
6. Gebhardt, F., Voss, A. Grather, W. & Schmidt-Belz. *Reasoning With Complex Cases*, Kluwer, 1997.
7. Gick, M. L. & Holyoak, K. J., Analogical problem solving, *Cognitive Psychology*, 12, 306—355, 1980.
8. Gick, M. L. & K. J. Holyoak. LISA: A computational model of analogical inference and schema induction. In G. W. Cottrell (Ed.), *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society* 352-357). Atlanta, GA: Lawrence Erlbaum Associates, 1996.
9. Hofstadter, D. & The Fluid Analogies Research Group, *Fluid Concepts and Creative Analogies*, Basic Books, New York, 1995.
10. Monaghan, J. M. & Clement, J. Use of computer simulation to develop mental simulations for understanding relative motion concepts. *International Journal of Science Education.* 21(9), 921—944, 1999.
11. Nersessian, N. J., *Faraday to Einstein: Constructing Meaning in Scientific Theories*. Kluwer, Dordrecht, pp. 68-93, 1984.
12. Pearce, M., Goel, A. K., Kolodner, J. L., Zimring, C., Sentosa, L., & Billington, R. Case-based design support: A case study in architectural design. *IEEE Expert: Intelligent Systems & Their Applications.* 7(5): 14-20, 1992.
13. Pedone, R., Hummel, J. E., and Holyoak, K. J. The use of diagrams in analogical problem solving. *Memory and Cognition*, 29:214—221, 2001.
14. Rehling, J. A. Letter Spirit (part two): Modeling Creativity in a Visual Domain 2001. Doctoral dissertation, Indiana University, 2001.
15. Yaner, P. W., Goel, A., Visual Case-Based Reasoning I: Memory and Retrieval. Proceedings of the *1st Indian International Conference on Artificial Intelligence*, 2003.
16. Winston, P. H. Learning and Reasoning by Analogy. Communications of the ACM, (23) 12, December 1980.