Emergent Properties and Structural Constraints: Advantages of Diagrammatic Representations for Reasoning and Learning

Kenneth R. Koedinger
Department of Psychology
Carnegie Mellon University
Pittsburgh, Pa 15213
Email: koedinger@psy.cmu.edu

This paper explores some general issues in diagrammatic reasoning using the domain of geometry as the primary example. Geometry has consistently played an important role as a model domain for studying diagrammatic reasoning and its relation to human and machine problem solving. In one of the early Artificial Intelligence papers, Gelernter (1963) showed how a problem diagram could help prune backward search in geometry theorem proving. More recently, in an article by Larkin and Simon (1987), which has been a real driving force of much of the current interest in diagrammatic reasoning, the authors turn to geometry as one source of evidence for their claims about the advantages of using diagrams in reasoning. Others have explored the role of visual images in geometric reasoning (Furnas, 1990; Kim, 1989) and learning (Suwa & Motoda, 1991). Without setting out to do so, we have found ourselves contributing to this literature as well. In trying to account for the abstract planning behavior of geometry experts, we discovered that a diagram-based representation provided a much better explanation of this behavior than the standard, sententially-based approaches to abstract planning (Koedinger & Anderson, 1990). We built a computer simulation called DC (the Diagram Configuration model) to demonstrate this.

I begin the paper by summarizing the work of Larkin and Simon (1987) in characterizing the advantages for diagrammatic representations. Then drawing upon the DC experience and the history of research in geometry problem solving, I elaborate on these issues and, in particular, discuss in more detail some advantages that were not fully addressed in Larkin and Simon. While there has been much said about the role of diagrams in problem solving, I also discuss their role in learning and in shaping the representations that result from this learning.

ADVANTAGES OF DIAGRAMS

Review of Larkin and Simon

In their paper "Why a Diagram is (Sometimes) Worth Ten Thousand Words", Larkin and Simon present a framework for understanding the role of diagrams in problem solving. In particular, they contrast the use of diagrammatic and sentential representations. A representation is composed of a *data structure* that stores states or steps of problem solving and a *program* that can interpret and modify this data structure. They define sentential and diagrammatic representations as follows (p. 68):

- A *sentential representation* has a data structure "in which elements appear in a single sequence" like the words and sentences in a text or the written steps in a problem solution.
- A *diagrammatic representation* has a data structure "in which information is indexed by two-dimensional location" like the components of a diagram.

The task is to explain when and why diagrammatic representations are more computationally efficient than sentential representations that contain the same information. Larkin and Simon (1987; p.98) list three reasons "why a diagram can be superior to a verbal description for solving problems":

- *Locality aids search*: "Diagrams can group together all information that is used together, thus avoiding large amounts of search for the elements needed to make a problem-solving inference."
- *Symbolic labels unnecessary*: "Diagrams typically use location to group information about a single element, avoiding the need to match symbolic labels."
- *Perceptual ease*: "Diagrams automatically support a large number of perceptual inferences, which are extremely easy for humans."

The next three sections cover these three points. Each section starts by reviewing the arguments made by Larkin and Simon and finishes with qualifications and/or extensions to these points.

Locality Aids Problem Search as well as Knowledge Search

Locality and Knowledge Search. A familiar strategy of high school geometry students is to record proof steps by marking the problem diagram (see Figure 1) as an alternative to writing them down in standard statement notation (see Table 1). Such an annotated diagram aids students in holding together information that they need to make further inferences. In contrast, information within a list of written statements may be visually separated and require search to identify.

For instance, to use the side-angle-side rule for inferring triangle congruence a problem solver must locate three congruence relationships – two between corresponding sides of the triangles and one between corresponding angles. In searching a list of statements

Prepared for the AAAI Spring Symposium Series, Reasoning with Diagrammatic Representations, Stanford, CA, March, 1992.

for these three relationships, one might need to consider numerous possible combinations of three statements that exist in the list. Consider the situation just before step 10 in Table 1. There are 9 already proven statements that might contribute to the next inference. If the the student is going to make a triangle congruence inference, she must find 3 of these statements that match with a triangle congruence rule (e.g., SSS, SAS, ASA, AAS). There are 9 choose 3 or 84 possible combinations to consider. In contrast, if these relationships are marked on a diagram (see Figure 1), one can quickly identify them since the side-angle-side configuration comes together in each triangle at a single vertex.

Table 1. Problem and solution in sentential form.

Problem statement in sentential form:

"Given $\overline{AC} \cong \overline{AD}$ and $\overline{AXB}$ bisects $\overline{CXD}$, prove that $\angle CBX \cong \angle DBX$."

Solution in sentential form:

Statements:	Reasons:
1. $\overline{AC} \cong \overline{AD}$	1. Given
2. $\overline{AXB}$ bisects $\overline{CXD}$	2. Given
3. $\overline{CX} \cong \overline{DX}$	3. Def-bisector (2)
4. $\overline{AX} \cong \overline{AX}$	4. Reflexive
5. $\triangle ACX \cong \triangle ADX$	5. SSS (1 3 4)
6. $\angle AXC \cong \angle AXD$	6. Corres-Parts (5)
7. $\overline{AB} \perp \overline{CD}$	7. Eq-Linear-Angs (5)
8. $\angle BXC \cong \angle BXD$	8. Eq-Linear-Angs (7)
9. $\overline{BX} \cong \overline{BX}$	9. Reflexive
10. $\triangle BCX \cong \triangle BDX$	10. SAS (3 8 9)
11. $\angle CBX \cong \angle DBX$	11. Corres-parts (10)

Locality and Problem Search. The example above illustrates the role of the diagram in aiding *knowledge search* – i.e., the search for applicable knowledge. The geometry diagram can also be used to aid *problem search* – i.e., the use of that knowledge to search for a problem solution (Newell, 1990). Diagrams can aid problem search or operator selection using the following heuristic.

> **Physical Distance-Reducing Heuristic**: An operation which reduces the physical distance between known and desired objects may also reduce the logical distance between them.

Greeno (1978) found that geometry students use this heuristic to solve a certain class of "angle-chaining" problems. Of course this heuristic does not always lead to the shortest logical solution. The problem in Figure 1 provides an example. After concluding that $\triangle ACX \cong \triangle ADX$ (see step 5 in Table 1), any three of the corresponding angles of these triangles might be proven congruent. The parts $\angle BXC \cong \angle BXD$ (step 6) are closest to the goal and thus, would be the ones selected by the physical distance-reducing heuristic. However, as the reader may have noticed, choosing $\angle CAX \cong \angle DAX$ can lead to $\triangle ACB \cong \triangle ADB$ and to a more direct solution. Nevertheless, in general this heuristic can lead to a solution more quickly. As an estimate, in this problem the probability of directly finding any one of the solutions is 50% higher using the heuristic (.17) than it is without it (.11).

Elaborations on Labelling Issue

Larkin and Simon's second advantage of diagrams is that they "eliminate" the need to keep track of and match symbolic object labels. This is an important consequence of the locality feature of diagrams (as such it is not really an independent advantage of diagrams). For example, in the diagram in Figure 1b, labels are not necessary to see that the angles marked with the 11 are part of the two triangles on the right in the diagram – this facilitates the inference from step 10 to 11. In contrast, making this inference in the sentential representation requires the use of point labels. To go from $\triangle BCX$ to $\angle CBX$, one has to notice that both contain the same three point labels **B**, **C**, and **X** – this guarantees that $\angle CBX$ is a part of $\triangle BCX$. Similarly, $\angle DBX$ is a part of $\triangle BDX$ because they share point labels.

An interesting twist on this issue is that the conventional labelling notation and formal rule set in geometry is not adequate to facilitate purely sentential reasoning. For example, from the label alone one cannot infer that $\angle CBX$ is a part of $\triangle BCA$ (the bigger triangle) – the angle refers to point **X** but not to **A** while the triangle refers to point **A** but not to **X**.

The claim that diagrammatic representations "eliminate" the need for object labels is too strongly stated. While the *point labels* in Figure 1b are not a necessary part of the diagrammatic solution (they were provided for reader to see the correspondence with Table 1), the *markings* on segments and angles are. Clearly these markings are a type of symbolic label and they serve the same purpose, "distal access" (Newell, 1990; p. 75), that the symbolic labels serve in a sentential representation. Despite the claim that diagrams group together information that is typically used together, some information that is needed together is not grouped together in the diagram. For example, the sides $\overline{AC}$ and $\overline{AD}$ are needed together as corresponding sides of triangles $\triangle ACX$ and $\triangle ADX$ in order to make the side-side-side inference in step 5. However, they are not grouped together in the diagram, at least they are not any more so than $\overline{AC}$ and $\overline{AX}$ which could conceivably be considered as corresponding parts of these two triangles. The single hash markings are a symbolism that directs the grouping together of $\overline{AC}$ and $\overline{AD}$.

Perceptual Ease and Emergent Properties

Perceptual Ease Through Practice. Larkin and Simon's third claim about the advantages of diagrammatic representations is that diagrams allow

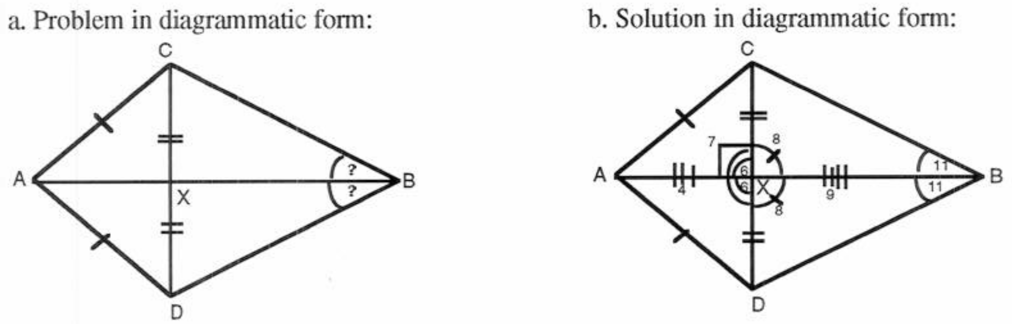

Figure 1. Diagrammatic versions of the problem and solution shown in Table 1. The numbered markings correspond with the steps in the sentential solution.

easy perceptual inferences to replace hard symbolic ones. This claim is based on an assumption that perceptual inferences are generally easier than symbolic inferences. While this assumption is probably true, it invites the question: why are perceptual inferences generally easier? In Koedinger and Anderson (1990), we expressed the feeling that "it is unlikely that perceptual inferences are somehow inherently easier (except in terms of the locality heuristic noted above)". Instead, we suggested that often perceptual inferences are more practiced than the corresponding symbolic inferences and this makes them seem easier. Clearly there are cases where symbolic inferences are easier than perceptual ones, for example, consider someone who is facile with symbolic logic but less practiced in using Venn diagrams. This person will be much faster doing syllogism problems symbolically rather than diagrammatically. Nevertheless, in many domains, like geometry and physics, it is clear that most people have had more prior experience with images than with formal notations and they are likely to find perceptual inferences in these domain easier.

Larkin and Simon present an example of a computer simulation in geometry theorem proving to illustrate the "perceptual ease" that is facilitated by diagrammatic representations. It is clear from the example, if not self-evident, that human's perform many perceptual inferences in this domain with much less effort (e.g., recognizing a triangle in a diagram) than corresponding logical inferences (e.g., recognizing a triangle from a list of sentential segment representations). However, the example provides no reason to believe that this relative ease is a function of the diagrammatic representation per se -- rather it may simply reflect our relative lack of experience in recognizing geometric figures from a sentential representation. The example provides no computational evidence that a diagrammatic representation would be more efficient than a sentential one. In fact, the computer code that simulated working with these two different representations was identical. Larkin and Simon's point, of course, was to argue (and convincingly so) that certain aspects of the process their code simulated are much easier for humans to do when given a diagrammatic representation than when given a sentential one. However, the example does not provide any clue as to *why* diagrammatic representations are easier for humans nor does it provide an indication of whether (or when) picking a diagrammatic representation over a sentential one will lead to a more efficient computational process (apart from the locality feature mentioned above).

Perceptual Ease Through Emergent Properties. Despite our earlier claim in Koedinger and Anderson (1990; see above), I now think there is another feature of diagrams besides locality that can lead to more efficient processing as well as provide an alternative to the practice explanation for "perceptual ease" in humans. In the process of drawing a diagram for a given situation, properties may emerge that weren't described in the original situation. These *emergent properties* are potential consequences of the given situation that can directly cue inferences that would require more costly indirect processing in a sentential representation.

This emergent-property feature of diagrams was hinted at in the following quote from Larkin and Simon:

> "The process of drawing the diagram *makes* these new inferences which are then displayed explicitly in the diagram itself." (p. 70)

They were referring to inferences mentioned above, like seeing **∠CBX** is the same as **∠CBA**, and modelled with their "perceptual inference rules". These inferences are based on the *topological* relations that are explicit in the diagrammatic data structure, but only implicit in the sentential data structure.

In addition to *topological* relations in diagrammatic data structures, there are also *geometric* relations that can lead to useful emergent properties. Whenever one draws a diagram one is making commitments, for example, about the sizes of segments and angles. These are not usually meaningful in any way and, in fact, they are *formally* irrelevant to validating the generalization to be proven. However, they can be used to guide problem search.

The Use of Emergent Properties in Geometry. Gelernter's (1963) Geometry Theorem Proving Machine was perhaps the first system to take advantage of these emergent geometric properties in diagrams. Gelernter used these properties as a heuristic to prune backward search. A statement (subgoal) generated by the backward application of a rule was compared against the diagram and if it looked false in the diagram it was rejected. This strategy works because in an accurately drawn diagram (i.e., one where the givens are true), all statements that could be proven will look true. In other words, the diagram can provide a counter-example to statements conjectured in backward reasoning.

Table 2. The input to DC for the problem above.

a. Problem diagram input (Figure 1a):

```
(line LINE-DXC points (D X C))
(line LINE-AXB points (A X B))
(line LINE-BC  points (B C)  )
(line LINE-DB  points (D B)  )
(line LINE-DA  points (D A)  )
(line LINE-AC  points (A C)  )
(point A x.coord 70  y.coord 83 )
(point D x.coord 123 y.coord 127)
(point C x.coord 123 y.coord 39 )
(point B x.coord 210 y.coord 83 )
(point X x.coord 123 y.coord 83 )
```

b Problem statement input (Table 1a):

```
(problem p1
   givens  ((congruent (segment A C)
                       (segment A D) )
           (bisector (segment A X B)
                     (segment C X D)))
   goal (congruent (angle C B X)
                   (angle D B X)) )
```

While Gelernter's program used the diagram to *prune* states that were *generated* from a sentential representation of the formal rules, essentially we have flipped these roles in our model of geometry problem solving (DC; Koedinger & Anderson, 1990). In DC, the diagram is used to *generate* states and formal knowledge of the domain is used to *prune* them. Putting more knowledge in the state generator leads to more efficient problem solving.

Table 3 shows the output of DC's diagram parsing and conjecturing (generation) processes for the problem in Figure 1. The input to this process is shown in Table 2a. The first part of Table 3 shows the results of DC's "diagram parsing", that is, the identification of objects that are constrained by the *topology* of the diagram. Diagram parsing is done using only the information in the line objects of Table 2a (the point positions are not necessary). This step is basically equivalent to Larkin and Simon's perceptual enhancement, it produces the objects (like segments, angles, and triangles) that are "perceptually obvious".

The next step "configuration conjecturing" makes use of *geometric relations* in the diagram, these result from the point positions specified in the point objects in Table 2b. DC uses this information to measure the sizes of segments and angles and then makes conjectures about various geometric configurations. These are tentative conclusions that indicate possibly (though not necessarily) provable states in the problem space for this problem. (It is only because of the compact nature of DC's representation, discussed below, that it is able to enumerate all of these states.)

DC recognizes 14 configurations in this problem and, in the process of recognition, connects them with other configurations based on overlapping parts. The result is a network, most of which is shown in Figure 2 (the remainder is not relevant to the solution of this particular problem). The task of the problem solver at this point is to search through this network to find a logically valid connection between the problem givens and the goal. DC's problem input is a combination of diagrammatic (Table 2a) and sentential (2b) forms – this is consistent with the way problems are usually presented in geometry textbooks. The sentential information about the problem givens and goals is used at this point in the problem solving where the search starts. Integrating the parsing, conjecturing and search processes would provide for a more accurate and efficient model, but by separating them we were able to identify the individual contribution of each. The search

Table 3. DC's diagram parsing and configuration conjecturing given Table 2a.

```
DIAGRAM  PARSING:
Segments: (SEG-AC SEG-DA SEG-DB SEG-BC SEG-AB SEG-AX SEG-XB SEG-DC SEG-DX SEG-XC)
Angles: (<CAB <BAD <CAD <BCD <DCA <BCA <CDB <ADC <ADB <DBA <ABC <DBC <CXB <BXD <DXA <AXC)
Adjacent supplementary angles: (SUPP-AXB-D SUPP-AXB-C SUPP-DXC-A SUPP-DXC-B)
Vertical angles at: (CROSS-X-ABCD)
Triangles: (TRI-ABD TRI-ADX TRI-ACD TRI-BDX TRI-BCD TRI-ABC TRI-BCX TRI-ACX)
CONFIGURATION  CONJECTURING:
Hypothesize bisected segment: (BISECTED.SEG-DC-AT-X)
Hypothesize perpendicular cross: (PERP-CROSS-X-ABCD)
Hypothesize bisected angle: (BISECTED.ANGLE-CAD-AT-B BISECTED.ANGLE-DBC-AT-A)
Hypothesize congruent pair of adjacent angles: (WP.ANGLE-ADB-C=ANGLE-BCA-D)
Hypothesize isosceles triangles: (ISOS-TRI-ACD ISOS-TRI-BCD)
Hypothesize right triangles: (RT-TRI-ADX-X RT-TRI-BDX-X RT-TRI-BCX-X RT-TRI-ACX-X)
Hypothesize shared-side congruent tris: (TRI-ABC=TRI-ABD TRI-ACX=TRI-ADX TRI-BCX=TRI-BDX)
```

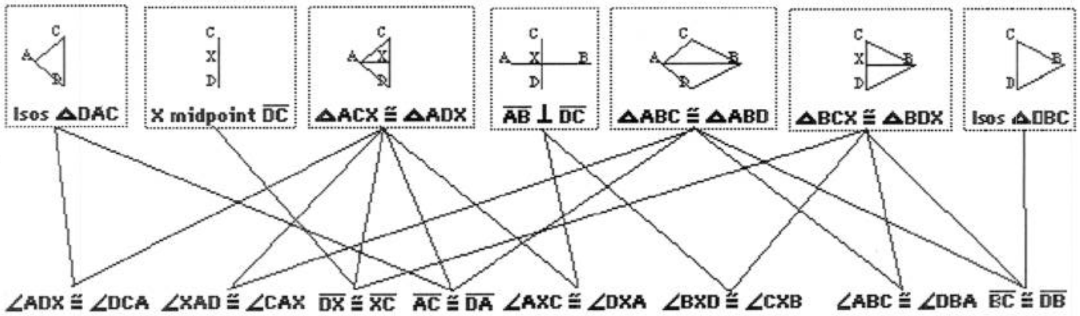

Figure 2. A portion of the network formed by DC's diagram parsing and configuration conjecturing components for the problem in Figure 1.

component is relatively easy. Most of the computational work is done in forming the network – increasing from about 60% to 96% of the total time as problems increase in difficulty. We'll say more about DC below, but the point to be made here is that the network formed as the result of configuration conjecturing is much smaller and much easier to search than the space of formally possible states that must be considered if the geometric relations in the diagram are not used. Some of the possible states in DC's problem space may not be provable – this will result when an overspecialized diagram is drawn, for example, if Figure 1a had been drawn so that **△ACD** is congruent to **△BCD** – however, these potential distractions are only a small subset of the distractions possible if geometric properties are not considered.

In general, because a diagram can contain properties beyond those needed to generate it, it can be used to convert a *nominally deductive* problem into an *inductive* one. Using the diagram one can induce (see) potential consequences of the given situation that aren't directly apparent in the verbal generalization to be proven. This is an important advantage of diagrams and of models more generally (Johnson-Laird, 1983).

USING STRUCTURAL CONSTRAINTS TO GUIDE LEARNING

As indicated in the introduction, we became interested in diagrammatic reasoning because it provided an answer to a problem we had: How could we model the abstract planning abilities we had observed in the behavior of the geometry experts. Using the verbal report methodology, we had found that skilled problem solvers make leaps of inference that skip over steps in a complete formal solution like the one shown in Table 1b. While coming up with a proof sketch, experts were most likely to mention steps like 5, 7 , and 10 and likely to skip the others. They would fill in the skipped steps after developing an abstract plan. As described in Koedinger and Anderson (1990), we found that neither the standard macro-operator nor abstract planning methods could provide a good explanation for the regularity in this step skipping behavior. Instead we found that expert's knowledge appeared to be organized around certain prototypical geometric configurations. These configurations group together clusters of geometric knowledge that can be cued by the diagram (as discussed above) and can lead to numerous inferences in a single step. These configurations were the basis of the DC model and we showed how they provided an accurate account of our step-skipping data and how they lead to a drastic reduction in the size of the search space. We have also used DC as the expert component of an Intelligent Tutor (Koedinger & Anderson, in press).

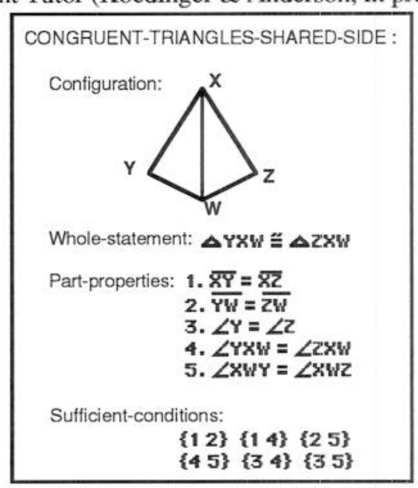

Figure 3. A diagram configuration schema. In the sufficient-conditions, {1 2} means that if the part-properties XY=XZ and YW=ZW are proven, all the properties of the schema can be proven.

Figure 3 shows a configuration schema. The configuration slot contains recognition knowledge that can pick out potential instances of the configuration in geometry diagrams. The top-row of the network in Figure 2 contains the 3 instances of the CONGRUENT-TRIANGLES-SHARED-SIDE configuration that occur in the diagram in Figure 1. The whole-statement slot indicates the geometry statement usually associated with this configuration. The part-properties slot indicate properties of parts of this configuration. They are known (i.e., can be proven) whenever the whole-statement is known or whenever one of the sufficient-conditions can be satisfied. These part-properties are shown in the bottom row of the network in Figure 2 and may overlap with other configuration schemas. The sufficient-conditions indicate subsets of the part-properties that are sufficient to prove the whole-statement and the remaining part-statements. This pattern completion nature of configuration schemas explains one way in which experts can skip steps.

These configurations are perceptual chunks (Chase & Simon, 1973) that experts have acquired from considerable experience in solving geometry problems. I separated out this section from the discussion on advantages of diagrams for reasoning, because while diagrams have some advantages for reasoning that are fairly independent of the nature of a problem solver's knowledge, the advantage made possible by perceptual chunks is very much dependent on the nature and organization of this knowledge. On the other hand, these perceptual chunks would not be learned if experts had not been working with diagrams. If, instead, they had been learning with a purely sentential version of geometry, they would not (at least, not easily) have acquired these perceptual chunks. One could achieve some level of expertise in the purely sentential version and might acquire macro-operators or abstractions that would make them steadily more effective and efficient. But, they would not achieve the efficiency facilitated by DC's configuration schemas.

We have argued that standard skill acquisition models cannot explain the acquisition of DC's schemas (Koedinger & Anderson, 1989, 1990). The difficulty arises because the basic domain rules (e.g., the rules in the reason column of Table 1) are organized in a very specific way within the DC model. The standard models provide no clear reason why this organization would evolve in contrast to the numerous other possibilities. A mechanism is needed that can take advantage of the structural constraints in geometry diagrams to guide how it organizes and chunks together the basic operators. This is a line of current research.

In a longer version of this paper[1], I give an example of how the techniques used in DC might be profitably generalized to physics problem solving.

CONCLUSION

This paper argues for the following advantages of diagrammatic representations (a modified and extended version of Larkin and Simon's list):

- *Locality advantages*. Pieces of information that need to be used together or logically connected are typically physically close together in diagrams. This feature of diagrams can be used to reduce, though not necessarily eliminate, knowledge search, problem search and the need for symbolic labels.
- *Emergent properties*. Diagrams can have both "psychological" and "computational" emergent properties. Psychological emergent properties are due to well-practiced perceptual inference knowledge that makes working with diagrams "easier" (for humans) than working with sentential representations. Computational emergent properties are due to geometric relations, for example, that can be used to drastically reduce the search space from the multitude of states that are syntactically possible in the sentential representation.
- *Structural constraints*. Whole-part relations in diagrams (and in models of the world more generally) can be used as a guide to efficient knowledge organization. A problem solver that is either given or can learn such a knowledge organization is likely to be more efficient than a problem solver who's knowledge is organized via sententially-based abstractions or macro-operators.

[1]This longer version is available from the author.

REFERENCES

Chase, W. G., & Simon H. A. (1973). The mind's eye in chess. In W. G. Chase (Ed.) *Visual Information Processing*. New York: Academic Press.

Furnas, G. W. (1990). Formal models for imaginal deduction. In *Proceedings of the Twelve Annual Conference of the Cognitive Science Society*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Gelernter, H. (1963). Realization of a geometry theorem proving machine. In E. A. Feigenbaum & J. Feldman (Eds.), *Computers and Thought*. New York: McGraw-Hill Book Company.

Greeno, J. G. (1983). Forms of understanding in mathematical problem solving. In S. G. Paris, G. M. Olson, & H. W. Stevenson (Eds.), *Learning and Motivation in the Classroom*. Hillsdale, NJ: Erlbaum.

Johnson-Laird, P. N. (1983). *Mental Models*. Cambridge, MA: Harvard University Press.

Kim, M. Y. (1989). Visual reasoning in geometry theorem proving. In *Proceedings of the International Joint Conference on Artificial Intelligence*, Detroit, MI, pp. 1617-1622.

Koedinger, K.R., & Anderson, J.R. (1989). Perceptual chunks in geometry problem solving: A challenge to theories of skill acquisition. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Koedinger, K. R., & Anderson, J. R. (1990). Abstract planning and perceptual chunks: Elements of expertise in geometry. *Cognitive Science*, 14, 511-550.

Koedinger, K. R., & Anderson, J. R. (in press). Reifying implicit planning in Geometry: Guidelines for model-based intelligent tutoring system design. In Lajoie, S. P. & Derry, S. J., (Eds.) *Computers as Cognitive Tools*. Hillsdale, NJ: Erlbaum.

Larkin, J., & Simon, H. A. (1987). Why a diagram is (sometimes) worth ten thousand words. *Cognitive Science*, 11, 65-99.

Newell, A. (1990). *Unified Theories of Cognition*. Harvard University Press, Cambridge, MA.

Newell, A., & Simon, H. A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.

Suwa, M., & Motoda, H. (1991). Learning abductive strategies from an example. In the working notes of the AAAI91 workshop on abduction.