Kurt's Thought Expanded

Noboru Matsuda
May 15, 2003

Is teaching strategy effective for learning problem solving? If so, how should it be taught?

Many task domains are deductive (monotonic problem solving) and covered here

- Geometry theorem proving
- Formal logic theorem proving
- Quantitative physics problem solving
- Qualitative physics problem solving
- Algebra word problems
- Law
- Problem solving in number theory, abstract algebra, calculus.
- Algebra equation solving

Expert problem-solving strategy is still poorly understood. (Ericsson & Lehmann, 1996)

- Organized around schemas, which are macro-operators. Angle modeled expert's problem solving performance with diagrammatic schema (Koedinger & Anderson, 1993).
- Often involves abstraction (leaving out details, such as numerical values for variables)
- Sometimes involves case-based reasoning (surface features cue recall of solution) (Chi, Feltovich, & Glaser, 1981)
- Sometimes involves search (problem must be novel, which is rare)
- Experts might perform equal amount of search as novices do, but with superior retrieval and appropriateness.
- Experts can be as bad as novices on unfamiliar problems.
- Entry order often does not reflect search (equations solved when entered FC) ????

The goal is to lead students to a mastery of problem solving; namely, make them capable of solve many problems in various types and difficulties. According to the theory of state-space search, students must learn following:

- Domain principles/concepts

- Heuristics as an aid to select appropriate domain principles
- Strategy to control search

"Heuristics" can be taught

- Schoenfeld showed that providing a list of heuristics along with a solution enhance learning, but not sufficient to achieve mastery. Students do need to learn strategy to control search (Schoenfeld, 1985). What he called "heuristics" partially corresponds to *strategy* in the current review. Example of such heuristic encourages to establish subgoals by decomposing a problem into a number of easier problems or by partly solving the problem.
- Some other heuristics seem to help appropriate domain principles to be retrieved and fired in a right context. An example of such heuristics is to provide a chance to expose (or envision) more instances of related propositions (could be referred to as *problem states*.
- We will not deal with heuristics in the current study.

Mastery of domain principles/concepts is probably a prerequisite for learning a strategy

- Empirical studies on cognitive load theory (Sweller) showed that example >> problem solving for novices, but problem solving >> example for intermediate students (Kalyuga, Chandler, Tuovinen, & Sweller, 2001).
- Self-explanation of examples can teach domain principles/concepts, but not strategy (except cases for CBR; citation ????). For 2-step arithmetic word problems, self-explanation did not improve learning (Mwangi & Sweller, 1998).
- Problem solving makes students confront lack of strategy and forces strategy learning. If we assume novices can't learn strategy until operators are mastered, then example + problem solving >> problem solving. If we assume that intermediates know operators and need to learn strategy, then problem solving >> example + problem solving.
- In physics, the major FC and BC heuristics for choosing an operator require applying it mentally and counting the unknowns in the resulting equation. This requires mastery of the operator. Also requires envisioning forces, etc. so must be at the level of a schema (????).
- The lit on part-task vs. whole task training seems to favor part-task, so that suggesting teaching single-operator problems before multiple-operator problems (Ash & Holding, 1990).
- Common wisdom is that mastery of subskills is required before teaching a skill that uses those subskills.

- But, Angle students seemed to have trouble expanding schema applications? 26% of 561 bottom-out hints (apparently including hints on schema application) did not work (Koedinger & Anderson, 1993, p.35)

Instructors seldom teach strategies

- A written proof doesn't show search processes. Proofs often written FC order, even if not discovered that way. Worked-out examples don't teach strategy.
- Instructors denigrate "turn the crank" strategies.
- Proof discovery considered a creative; Polya (teaching heuristics) is best we can do.

Strategies are supposed to have two benefits. One is that they prevent students from wasting time by making entries that have no connection to a solution tree. The other is that they prevent students to get stuck for strategies always tell a "legal" step that can be done next.

- Strategic hint enhanced students' performance on hard problems, but not easy ones (Scheines & Sieg, 1994).
- In the Related Rates Tutor, the goal reification and guide on backward chaining facilitated learning but the effect size was quite small (Singley, 1990).
- GIL vs. exploratory GIL showed the time-achievement tradeoff (Reiser, Copen, Ranney, Hamid, & Kimberg, 1994).
- In principle, all what we need is backward chaining. However, possibly due to working-memory capacity, we – human problem solvers – can't aware all possible backward applications. Hence, we need to learn bidirectional search. The study on CMU proof tutor supports this (Scheines & Sieg, 1994).

Some tutoring systems do teach strategies.

- Ms. Lindquist explicitly teaches strategies for discovering algebraic equations for word problems by providing guidance in BC (Heffernan & Koedinger, 2001, submitted ??). Its effectiveness is not reported anywhere…
- PACT tutors are less explicit, but do support Ms. Lindquist's inductive strategy. The basic idea is to implement a set of production rules that articulate problem-solving strategies (e.g., subgoaling for a particular goal).
- CPT requires students to enter only FC or BC rule applications. Teaching in bidirectional strategy is superior to any one-way strategy (Scheines & Sieg, 1994).

- ANGLE requires following specific proof strategy including configuration parsing and matching (Koedinger & Anderson, 1993). Learning gain? Must read
- The Related Rates Tutor (Singley, 1990) explicitly teaches BC by telling what to do next for each step of BC.
- GIL teaches means-ends analysis by providing next MEA step (FC achieving a goal, BC, or a legal FC).
- Pyrenees ????

Some tutoring systems teach strategies implicitly. They may block entering inappropriate (either false or not a part of proof) propositions. In other words, they let students enter propositions that don't agree with the strategy. When asked to suggest a step, they may pick one according to a strategy, and the hints may reveal the strategic reasoning (e.g., a chain of goals), but the system doesn't enforce attention to the strategic instruction.

- Andes1 (Gertner & VanLehn, 2000) ????
- Andes2 hints a partial strategy, but doesn't require following it (VanLehn et al., 2002) ????
- Angle allows students to enter redundant propositions, but doesn't take them into account when providing next-step hint (citation ?).
- Why2 ????
- GeoLog ????

Some tutoring systems offer students "exploratory" learning environment, but most of the time they obey the principle of time-achievement tradeoff; the number of problems solved does matter, but not how they were solved.

- Experiment with GIL shows that exploratory group (allow illogical and incomplete input with on-demand feedback) took longer learning time than guided group (strict MTT with immediate feedback), but the group ended up with the equal learning gain as MTT group (Reiser et al., 1994). Even the free group (no exercise problems provided. Students can test and run their own program) acquired comparable coding skills (p.30). Furthermore, the exploratory group showed better performance in identifying bugs (no group difference in *repairing* bugs).
- (Charney & Reder, 1986) showed that pure problem-solving (no guidance hence exploratory) is more difficult form of training than guided practice, but it produces better performance at test.
- These studies suggest that it might be the number of applications of domain principles that affects learning, not the number of problems solved, which in geometry corresponds to the number of postulate application. Teaching strategy does not matter? – Maybe it does; the strategy helps

students solve more problems in a fixed time, which in turn let them practice more domain principles.

- Number of postulates applied (no matter if they are part of a solution or not) must be counted. Singley's study claimed that the goal blackboard prevented students to follow a garden path. But, it also means that they are blocked to expose more operator applications?

How to guide true/relevant but illogical input is not clear. Pam showed (proof-talk) that when allowing students to enter any true/relevant propositions, it is technically possible conduct a dialogue guiding them to connect it up to FC or BC fringe. But, it is not clear what such a dialogue should say.

- Her prototype says "How can you justify that <most recent entry>?" of "Given that <earlier entries>, how can you prove <most recent entry>?" or "What follows from <most recent entry>?" That is, it targets a node just above or below the most recent node and prompts the inference leading from the most recent node to the target node.
- In qualitative physics, propositions may be mentioned by the student just because they seem true, so it makes sense to prompt for justifications. In geometry, some angles and segments may look equal, so students might put those propositions down, but they should know that they are only conjecture until proved.

How to make the best user of redundant (true/irrelevant) input is another controversial question.

- Some tutors just provide negative acknowledgement by saying "No. It's not an appropriate input."
- Some tutors accept a redundant input, but disregard it at the time to provide a hint.
- Redundant input may be appreciated in a sense that it is evidence that the student know how to apply domain principle.

Some tutors ask student to produce a solution that is different from the educational objectives

- GIL, the *LISP programming* tutor, asks student to complete a graphical structure of a target LISP function, not a LISP program
- GTP and Angle, the *geometry proof* tutor, ask students to build a proof tree, not a traditional proof
- It might be questionable that those students can actually produce LISP function and proofs.

Reification is a commonly used technique for scaffolding

- Many tutoring systems reify a goal structure. Relation Rates Tutor (Singley, 1990), Angle, CPT, Geometry Tutor, …

- What is the "goal structure"? It shows a *prescriptive* aspect of a domain principle; namely, it shows how to achieve a certain goal. So, it's not necessarily represented in the tree structure, rather the relationship between premise and conclusion must be emphasized (the tree structure does emphasize it). However, the tree structure also represents a goal stack, i.e., what's left to be done.
- Reification of domain principles. Angle has list of iconic postulates.
- Some reifies of application of domain principles. Angle

Scaffolding should be faded

- Conversion from descriptive (declarative) principles to prescriptive (procedural) rule. "If a pair of triangles is congruent, then corresponding elements are equal" ? "To prove angles are equal, prove a pair of triangles is congruent."
- Operator selection: Related Rates Tutor provides BC. The basic idea is that the tutor simply shows next available operator (either only the one that should apply next or a set of "legal" operators) and the student must figure out how to apply (instantiate) it to the problem state.
- Operator application: Angle
- The above three can be defined hierarchically.
- Strategy
- Goal stuck: Many ITS with goal reification

Student's input

- Restricted input
- Free input

Self-explanation

- Reification
- Self-explanation
- Hinting

Reference:

Ash, D. W., & Holding, D. H. (1990). Backward versus forward chaining in the acquisition of a keyboard skill. *Human Factors*, *32*(2), 139-146.

Charney, D. H., & Reder, L. M. (1986). Designing interactive tutorials for computer users. *Human Computer Interaction, 2*(4), 297-317.

Chi, M. T. H., Feltovich, P. J., & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science, 5*, 121-152.

Ericsson, K. A., & Lehmann, A. C. (1996). Expert and exceptional performance: Evidence of maximal adaptation to task constraints. *Annual Review of Psychology, 47*, 273-305.

Gertner, A. S., & VanLehn, K. (2000). Andes: A coached problem solving environment for physics. In G. Gauthier & C. Frasson & K. VanLehn (Eds.), *Intelligent Tutoring Systems, Proceeding of 5th International Conference on ITS* (pp. 133-142). Berlin, Heidelberg: Springer.

Heffernan, N. T., & Koedinger, K. R. (2001, submitted). Expanding the Model-Tracing Architecture: A 3rd Generation Intelligent Tutor for Algebra Symbolization. *Journal of Artificial Intelligence in Education*.

Kalyuga, S., Chandler, P., Tuovinen, J., & Sweller, J. (2001). When problem solving is superior to studying worked examples. *Journal of Educational Psychology, 93*(3), 579-588.

Koedinger, K. R., & Anderson, J. R. (1993). Reifying implicit planning in geometry: Guidelines for model-based intelligent tutoring system design. In S. P. Lajoie & S. J. Derry (Eds.), *Computers as cognitive tools* (pp. 15-45). Hillsdale, NJ: Lawrence Erlbaum Associates.

Mwangi, W., & Sweller, J. (1998). Learning to solve compare word problems: The effect of example format and generating self-explanations. *Cognition and Instruction, 16*(2), 173-199.

Reiser, B. J., Copen, W. A., Ranney, M., Hamid, A., & Kimberg, D. Y. (1994). *Cognitive and motivational consequences of tutoring and discovery learning* (Technical Report #54). Evanston, IL: The Institute for the Learning Sciences, Northwestern University.

Scheines, R., & Sieg, W. (1994). Computer Environments for Proof Construction. *Interactive Learning Environments, 4*(2), 159-169.

Schoenfeld, A. H. (1985). *Mathematical Problem Solving*. Orlando, FL: Academic Press.

Singley, M. K. (1990). The Reification of Goal Structures in a Calculus Tutor: Effects on Problem-Solving Performance. *Interactive Learning Environments, 1*(2), 102-123.

VanLehn, K., Lynch, C., Taylor, L., Weinstein, A., Shelby, R., Schulze, K., Treacy, D., & Wintersgill, M. (2002). Minimally invasive tutoring of complex physics problem solving. In S. A. Cerri & G. Gouarderes & F. Paraguacu (Eds.), *Proceedings of the 6th International Conference on Intelligent Tutoring Systems* (pp. 367-376).