Understanding Algebra Equation Solving Strategies

Technical Report PCG-2

Kurt VanLehn and William Ball

Departments of Psychology and Computer Science
Carnegie-Mellon University
Schenley Park,
Pittsburgh, PA 15213 U.S.A.

6 July 1987

Running head: Algebra strategies

This research was supported by the Personnel and Training Research Programs, Psychological Sciences Division, Office of Naval Research, under Contract number N00014-86-K-0349. Approved for public release; distribution unlimited.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; Distribution unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) PCG-2	5. MONITORING ORGANIZATION REPORT NUMBER(S) Same as Performing Organization
6a. NAME OF PERFORMING ORGANIZATION Carnegie-Mellon University 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Personnel and Training Research Office of Naval Research (Code 1142 PT)
6c. ADDRESS (City, State, and ZIP Code) Department of Psychology Pittsburgh, PA 15213	7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy St. Arlington, VA 22217-5000
8a. NAME OF FUNDING / SPONSORING ORGANIZATION Same as Monitoring Organization 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-86-K-0349
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS 442a558-02/12-15-86 PROGRAM ELEMENT NO. N/A; PROJECT NO. N/A; TASK NO. N/A; WORK UNIT ACCESSION NO N/A
11. TITLE (Include Security Classification) Understanding Algebra Equation Solving Strategies	
12 PERSONAL AUTHOR(S) Kurt VanLehn & William Ball	
13a. TYPE OF REPORT Technical 13b. TIME COVERED FROM 86Jun15 TO 87Oct31	14. DATE OF REPORT (Year, Month, Day) 87 July 6 15. PAGE COUNT 13
16 SUPPLEMENTARY NOTATION	
17 COSATI CODES FIELD / GROUP / SUB-GROUP	18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number) Algebra, equation solving, problem space, problem solving
19. ABSTRACT (Continue on reverse if necessary and identify by block number) See reverse side.	
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION
22a NAME OF RESPONSIBLE INDIVIDUAL Susan Chipman	22b TELEPHONE (Include Area Code) (202) 696-4322 22c. OFFICE SYMBOL 1142 PT

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Unclassified

Abstract

A task analysis of linear algebraic equation solving is presented. The problem space is shown to have an elegant mathematical form. Several strategies for searching the problem space are delineated, and their properties discussed. The forward search strategy, which appears to be the one most commonly taught in high-school textbooks, tends to generate non-optimal solution paths. An operator-subgoaling search strategy tends to generate shorter paths. Bundy and Welham's waterfall loop strategy is shown to be a variant of operator-subgoaling that is more amenable for use by humans. This task analysis suggests that the waterfall loop strategy may be better for teaching to high-school students than the forward search strategy.

1. Introduction

This brief note presents an analysis of a small portion of high school algebra, the solving of linear equations in one variable. The analysis is based on the formal properties of the task, rather than data from human subjects. Two results are presented. First, the structure of the task domain is uncovered, and shown to have some elegant properties. Second, there is suggestive evidence that forward search, which appears to be the strategy most commonly taught in high school algebra, is a less efficient search strategy than operator subgoaling, a strategy based directed on the structure of the task domain, in that it tends to generate longer solution paths. This suggests that operator subgoaling may be a better strategy for teaching high school algebra students. However, operator subgoaling seems to require more cognitive resources of the student than forward search. Bundy and Welham's (1981) waterfall loop strategy, which is a type of operator subgoaling, is shown to offer reduced requirements for cognitive resources. It thus combines the advantages of short solution paths with low cognitive load.

Although these results are suggestive, empirical work is needed in order to build a case for changing the pedagogical practices of high school algebra. A tenuous, but still interesting conjecture is that teaching students the *structure* of the solution space, as described herein, may lead them to a better *understanding* of the process of solving algebra equations.

2. The problem space of simple algebraic equation solving

Solving an algebra equation can be viewed as search in a problem space (Newell & Simon, 1972). A problem space is defined by a set of states, a set of operators for moving from one state to another, an initial state, and a description for the desired final state. For algebra, a state is just an algebraic equation, and a operator is just an algebraic transformation, such as subtracting a term from both sides of the equation. The initial state is the given equation, e.g., $6-5(x+3)+7x = 3-x$. A final state is any equation that has just one occurrence of the variable, and the variable is isolated, that is, it stands alone on the left or right side of the equation.

Different initial states (i.e., different equations to be solved) engender different specific problem spaces. However, all the problem spaces in this task domain have the same basic topological form. This section discusses that form.

The form is hierarchical. We will define a hierarchy of state types, such that a state of type N is an equivalence class of states of type N-1. The lowest state type, state type 1, consists of the actual

equations. Thus, $1+x = 3$ and $x+1 = 3$ are distinct type 1 states. A type 2 state is an equivalence class of type 1 states that can be reached by the algebraic transformations for communtativity, associativity, arithmetic combination, reversing the sides of an equation, and simplifying a double unary minus. Thus, the following equations are all in the same type 2 class:

$x+1 = 30/3$

$1+x = 30/3$

$1+x = 10$

$1+x = -(-10)$

$10 = 1+x$

$10 = x+1$

$11-1 = x+1$

$100^{-1/2} = x+1$

Because there are infinitely many ways to express numbers as arithmetic expressions, there are infinitely many type 1 states in a type 2 state.

Type 3 states are defined as the equivalence class of type 2 states under the algebraic transformations that "do the same thing" to both sides of the equations, such as adding the same term to both sides of the equation. Such equivalance classes have an interesting structure. It is easiest to discuss it with the aid of several simple examples.

Figure 1 shows eight type 2 states that constitute a type 3 state. They are arranged in a cube in order to show the relationships among them. The edges that are parallel to the horizontal axis are represent the operations of multiplying or dividing by a. The vertical axis edges represent multiplying or dividing by b. The remaining edges represent multiplying or dividing by c. The diagonals of the cube (not drawn) correspond to inverting both sides of the equation.

Figure 2 displays another type 3 state, where the equations are related by adding and subtracting from both sides. The edges represent adding and subtracting by, respectively, a, b or c. The diagonals represent negating both sides of the equation.

In general, all equations formed from three atomic subexpressions and an invertible binary operator will engender a type 3 state that has either type 2 states arranged in a cube. However, if the subexpressions are not atomic, but are themselves expressions, then a type 3 state consists of two or more cubes that

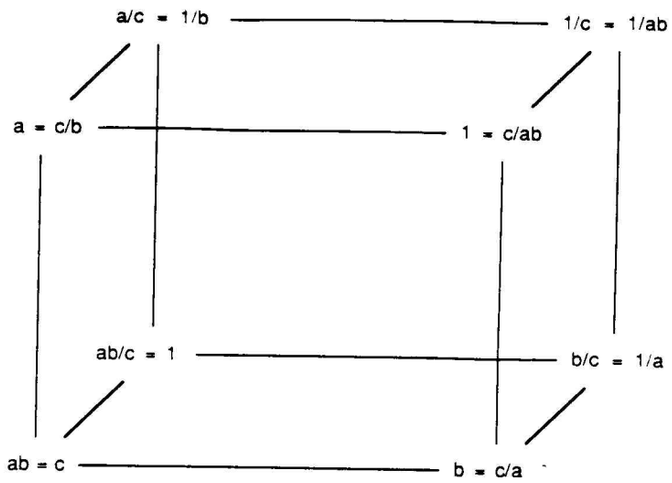

Figure 1: Type 3 state for multiplication and division

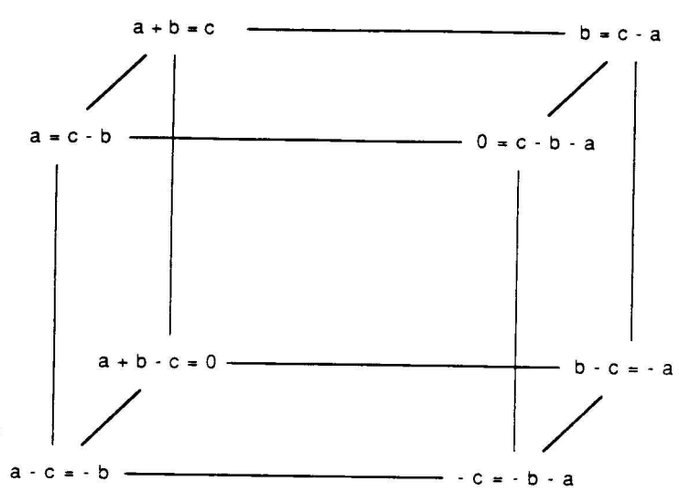

Figure 2: Type 3 state for addition and subtraction

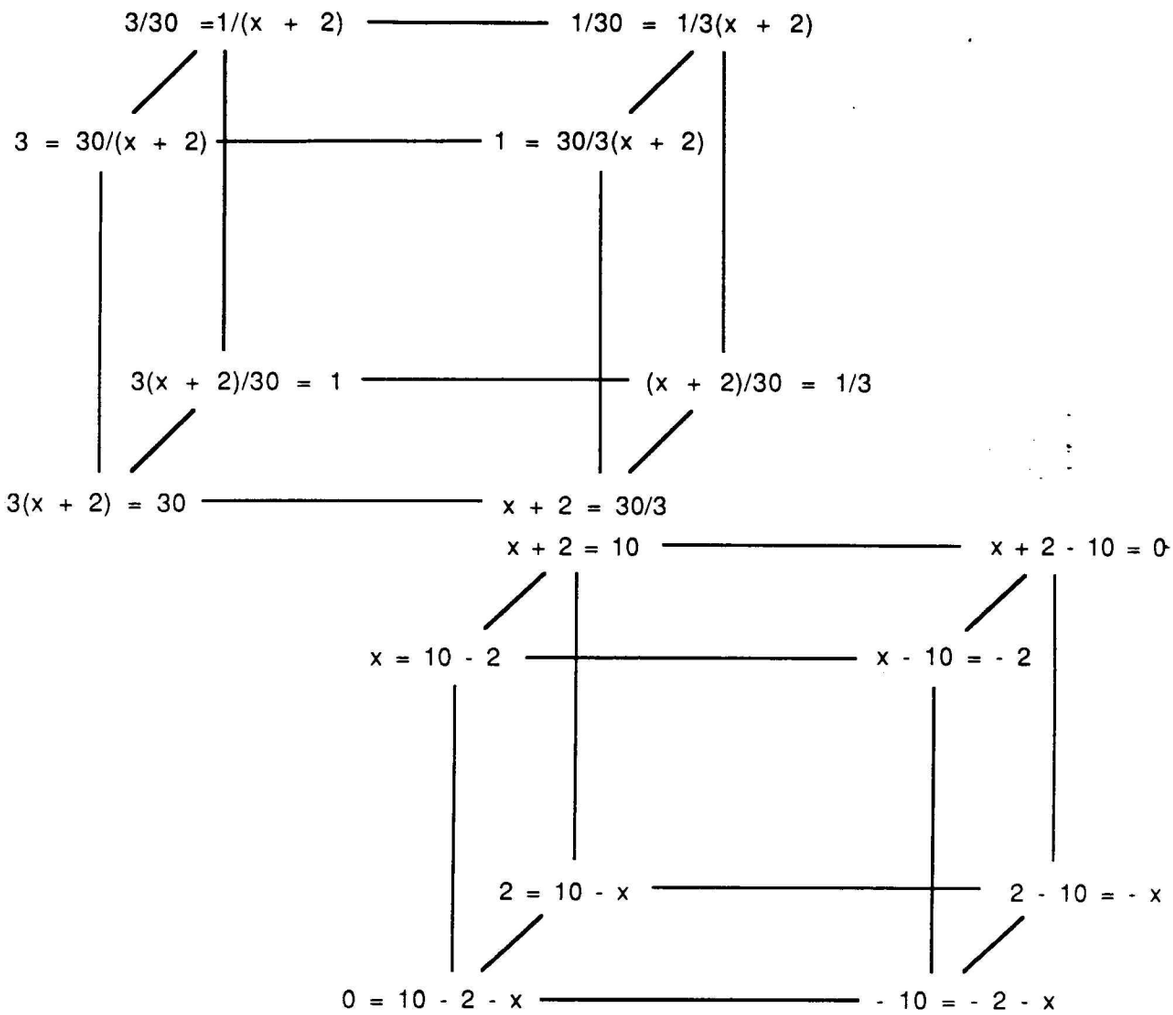

Figure 3: Type 3 state for the equation 3(x+1) = 30.

share vertices. Figure 3 illustrates such a state for the equation $3(x+1) = 30$. The upper cube treats the subexpression $(x+1)$ as atomic, while the bottom cube treats 3/30 as atomic. The multiplication cube shares the vertex that stands for the equation $x+1 = 3/30$ with the addition cube. In general, there are three vertices in every cube that can be shared, viz. those where a subexpression that is treated as atomic relative to the cube's operations appears isolated on one side of the equality. When a subexpression is isolated, its operator is the top operator on one side of the equation, and thus can found a new "do it to both sides" cube.

In principle, one can add (or multiply, etc.) any expression to both sides of an equation. Thus, adding $234y$ to both sides of $2(x+1) = 30$ is legal. However, most algebra equations can be solved by applying both-sides operations using only subexpressions that appear in the equation. If we restrict the problem space by only allowing the both-sides operators to use subexpressions from the equation, then a type 3 state always consists of a set of one or more cubes, connected by shared vertices. With this restriction, a type 3 state has the following properties:

- All its constituent type 2 states are equations with exactly the same atomic terms (i.e., modulo arithmetic evaluation, which merges number together).
- Any atomic term can be isolated by operations that stay inside the type 3 state.

These properties follow directly from the fact that each operation neither destroys terms nor creates terms, and that all operations are invertible, given that the equations are linear equations.

These two properties together imply that if any equation in a type 3 state has a single occurrence of the variable, then they all do, and furthermore, that one of the type 2 equations has the variable isolated on one side of the equation. Thus, a type 3 state is a "final" state if it contains an equation with a single occurrence of the unknown. For example, the equation $3(x+1) = 30$ corresponds to a final type 3 state; mere "both sides" operations are all that is required to convert it to the desired form.

To put the property more generally, all the equations in the type 3 state have the same number of occurrences of the unknown variable. A type 3 state can be assigned a heuristic value equal to the number of occurrences of the unknown, and this value can be used to guide the search among type 3 states by always choosing operations reduce this value. This strategy is discussed further below.

The only way to change the set of atomic expressions in an equation is to use some form of distribution or its converse, combination. These operations are illustrated below:

$x(a+b) = c \leftarrow\rightarrow ax+bx = c$

$c = (a+b)/x \leftarrow\rightarrow c = a/x+b/x$

$x^a x^b = c \leftarrow\rightarrow x^{a+b} = c$

These distribution operations form the type 3 state transitions. Thus, the whole problem space for linear[1] algebra equation solving can be reduced dramatically to a problem space of type 3 states connected by type 3 operators, with the final type 3 state having just a single occurrence of the variable. This idea forms the basis of the operator subgoaling strategy, which is discussed in the next section.

3. Two search strategies

This section discusses two search strategies, then compares them. The first strategy derives from the problem space analysis presented above. The idea is to search through the type 3 states using a simple hill climbing strategy -- move to an adjacent type 3 state that minimizes the number of occurrences of the variable -- then search through the final type 3 state to find a final type 2 state.

Although this sounds quite simple, it is complicated by the fact that the combine-term operations, which are the operators used to move between type 3 states, require that the terms to be combined are cousins in the expression tree. That is, when the expression is viewed as an operator tree (see figure 4), the two occurrences of the variable must be direct descendents of sibling nodes. In the equation $3x+4x+4 = 7$ it is possible to coallesce the occurrences of the variable, but in the equation $3x+4(x+1) = 7$ it is not possible apply the combine-terms operation. Thus the strategy requires *operator subgoaling*, wherein the solver adopts the new goal of transforming the equation into a form suitable for applying the combine-terms operator. In this case, the subgoal is to transform the subexpression $4(x+1)$ into a sum with x contained in one of its terms. This can be accomplished by applying the distribution operator, which corresponds to a transition between two adjacent type 3 states, both of which have same heuristic value. In some cases, the subgoal can be accomplished by staying inside the type 3 state, as when $3x = 7-4x$ is transformed to $3x+4x = 7$. Such subgoaling can become rather complicated. Because the dominant form of activity is operator subgoaling, this strategy is named *operator subgoaling*.

As mentioned earlier, final states can be specified as a conjunction of two properties, (1) there is one occurrence of the variable, and (2) it occurs isolated on the left or right side of the equation. The operator

[1]The analysis can probably be extended to a much larger class of equations. Bundy and Welham's (1981) equation solving system, which will be shown later to use a special case of this strategy, can solve rational polynomials containing trigometric and hyperbolic functions.

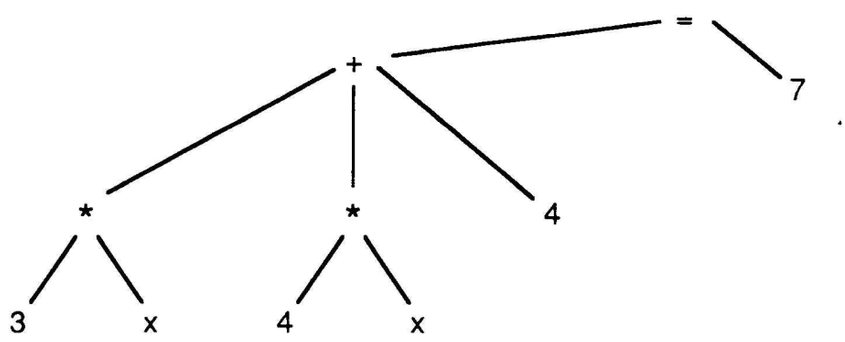

Figure 4: Expression tree for the equation 3x+4x+4=7. The "*" means multiply.

subgoaling strategy achieves the single-occurrence goal first, then works on the isolation goal. The search strategy that seems to be taught in high school emphasizes the isolation subgoal. In the textbooks we have examined, students are taught to clear radicals, fractions, and parentheses first, then combine terms. (See figure 5 for one popular textbook's description of the strategy it teaches). This isolate-then-combine strategy has the advantage that it is easily implemented as a visually-cued forward search. The search heuristics are rules such as "If you see some parentheses, then use the distribution operator to remove them." Because the heuristics are cued by visual features of the equation, they may be easier to remember.

However, the forward search strategy leads to inefficient solutions in some cases:

Forward Search Solution	Optimal Solution
$3(x+2) = 30$	$3(x+2) = 30$
$3x+6 = 30$	$x+2 = 10$
$3x = 24$	$x = 8$
$x = 8$	

The optimal solution path in this case would be generated by the operator subgoaling strategy. The forward search's strategy is non-optimal because it moves out of the initial type 3 state, which is also a final state, and into an adjacent type 3 state, which is also a final state. In this case, it is better to stay in the initial type 3 state. Here are some more cases where the forward search strategy does poorly:

SOLVING AN EQUATION HAVING ONE VARIABLE

1. Clear the equation of fractions, if any, by multiplying both members by the L.C.D. of all fractions in the equations.

2. Remove parentheses.

3. Clear the equation of decimals, if any, by multiplying both members by the appropriate power of 10.

If it is a first-degree equation,

4. Collect all terms containing the variable so they are in the left member. Collect all other terms in the right member.
5. Simplify both members.
6. Divide each member by the coefficient of the variable.
7. Check your result by replacing the variable in the original equation.

If it is a second-degree equation,

4. Collect the terms so they are in the left member. The right member should be zero.
5. Simplify the left member.
6. Factor the left member.
7. Set each factor containing the variable equal to zero and solve each resulting equation.
8. Check your results by replacing the variable in the original equation.

Figure 5: A procedure from a popular high school textbook, Welchons et al. (1981), pg. 419.

$4(3b+4)-3x = 147$	Removing the parentheses is unnecessary
$(3b+4)(3x+7) = -4-3b$	Shortest path is to divide by $3b+4$
$27[3x+7x+9] = 102$	Shortest path is to divide out the 27 first

We believe that it can be shown that the operator subgoalling strategy always produces shorter solution paths than forward search, or a path of equal length. This belief is based mostly on our inability to find a counterexample. Further work is needed on this important question.

4. Comparing the two strategies

For human solvers, it is important to find shorter paths, but not so much because it saves time, but rather because it reduces the chance of error. Experienced solvers make most of their errors during the execution of operations, as opposed to using incorrect or inappropriate operations (Lewis, 1981). The fewer the operations needed to achieve a solution, the less chance of error.

However, shortness of path is not the only relevant criterion upon which a search strategy should be chosen. The strategy should be easy to use and easy to learn. The operator subgoaling strategy may not be particularly easy to use, because it seems to require a goal stack. That is, the solvers must remember what goal they were working on so that they can resume working on it when they get done with the subgoal. The extra memory load of a maintaining a goal stack may make the operator subgoaling strategy more difficult to use than the forward search strategy, which requires no goal stack because its selection of operators is determined entirely by the current state.

The waterfall loop strategy (Bundy & Welham, 1981) offers the best of both strategies. It is essentially an operator subgoaling strategy, which means that it tends to generate optimal solution paths, but it is implemented by several heuristics that are driven almost entirely by the current state, thus minimizing potential memory load. The waterfall loop strategy has three "meta" rules:

- *Isolation:* If the equation has just one occurrence of the variable, then apply a both-sides operator appropriate to the arithmetic operation that is the root of the expression tree on the side of the equation containing the variable.
- *Combination*: If the equation has two occurrences of the variable that are cousins in the tree, then combine them.
- *Attraction*: If the equation has multiple, non-cousin occurrences of the variable, select two that are nearest in the tree, and apply a transformation that will make them nearer.

The first of these meta-rules that matches the equation is fired, then the loop repeats on the new

equation. That is, control falls through to the rule that matches, then loops back.

The waterfall loop strategy has the same goal structure as the operator subgoaling strategy. Isolation and combination are the top level goals of algebra equation solving, and attraction is a subgoal of combination. The waterfall loop differs from operator subgoaling in that it uses no goal stack as temporary state for its processing. It is driven entirely by the appearance of the equation. So it too is a visually cued, forward search strategy. But its *design* makes it a form of operator subgoaling.

The waterfall loop therefore combines the best properties of both forward search and operator subgoaling. It tends to generate optimal solution paths, and it is visually cued. The differences between it and the usual procedure taught in schools is that its cues concern the number of occurrences of the variable and their relative position in the equation tree. The forward search procedure is cued by the presence of large features, such as parentheses and radicals.

Obviously, these comments about ease of use must be viewed as suggestive only. Experimental work, perferably with both expert and novices human solvers, is needed to compare the two strategies.

5. Suggestions for further research

This brief note, although sparse on results, opens a number of interesting avenues for research. Two have been mentioned already: a formal demonstration of the optimality of the operator subgoaling strategy over the forward search strategy, and an empirical demonstration of its superior ease of use. Similarly, we need to experimentally compare the learnability of the two strategies.

A possibly more important question is whether this new view of algebraic equation solving as simple hillclimbing in the type 3 problem space allows students to truly *understand* the task domain. Certainly, we feel that we understand algebraic strategies better for having understood the structure of the problem space. Perhaps this view will help the students as well.

To put the suggestion in more concrete form, suppose one built an algebra equation solving system along the lines of Algebraland (Brown, 1983) that used a menu-driven interface to allow the student to select operations, which the system would then apply. Algebraland keeps track of the path the student follows and displays it; if the student backs up, then the display is a tree, otherwise it is a path. Brown claims that this may facilitate the acquisition of improved search strategies. This claim is consistent with research by Anderson and his colleagues (Anderson, Boyle & Yost, 1985; Anderson, Boyle & Reiser,

1985), which shows that similar displays of solution trees seems to help geometry students learn strategies for finding proofs in plane geometry. The basic message from both sets of researchers is that displaying the solution path in a way that emphasizes its tacit structure helps students learn a search strategy based on that structure. Now, suppose that we displayed the search of an algebra student in a manner similar to the cubes of figures 1, 2 and 3. We conjecture that this display will help students come to understand algebra strategy in a new, more beneficial way.

References

Anderson, J.R., Boyle, C.F. & Reiser, B. (1985). Intelligent tutoring systems. *Science*, *228*(4698), 456-462.

Anderson, J. R., Boyle, C., & Yost, G. (1985). The geometry tutor. In *Proceedings of Ninth International Joint Conference on Artificial Intelligence.* Los Altos, CA: Morgan-Kaufman, 1-7.

Brown, J.S. (1983). Process versus product: A perspective on tools for communal and informal electronci learning. In *Report from the learning lab: Education in the electronic age.* New York, NY: Educational Broadcasting Company.

Bundy, A., and Welham, B. (1981). Using meta-level inference for selective application of multiple rewrite rule sets in algebraic manipulation. *Artificial Intelligence*, *16*, 189-211.

Lewis, C. (1981). Skill in algebra. In J. R. Anderson (Ed.), *Cognitive Skills and their Acquisition.* Hillsdale, NJ: Lawrence Erlbaum.

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

Welchons, A.M., Krickenberger, W.R., Pearson, H.R., Duffy, A.G., M& MacCaffery, J.M. (1981). *Algebra: Book 1.* Lexington, MA: Ginn.