COGNITIVE SCIENCE **17**, 497–529 (1993)

Problem-Solving Restructuration: Elimination of Implicit Constraints

JEAN-FRANÇOIS RICHARD
SÉBASTIEN POITRENAUD AND CHARLES TIJUS
University of Paris 8. France

A general model of problem-solving processes based on misconception elimination is presented to simulate both impasses and solving processes. The model operates on goal-related rules and a set of constraint rules in the form of "IF (state or goal), DO NOT (Action)" for the explicit constraints in the instructions and the implicit constraints that come from misconceptions of legal moves. When impasses occur, a constraint elimination mechanism is applied. Because successive eliminations of implicit constraints enlarge the problem space and have an effect on planning, the model integrates "plan-based" and "constraint-based" approaches to problem-solving behavior.

Simulating individual protocols of Tower of Hanoi situations shows that the model, which has a proper set of constraints, predicts a single move with no alternative on about 61% of the movements and that protocols are quite successfully simulated movement by movement. Finally, it is shown that many features of previous models are embedded in the constraint elimination model.

Tower of Hanoi problems, missionaries–cannibals problems, water–jugs problems and a variety of Nim games are classical puzzle-solving situations that have received a great deal of attention because they do not involve domain-specific knowledge, and can hence be used to investigate basic cognitive mechanisms such as search mechanisms and decision-making mechanisms. For such puzzle-like situations, research on simulating problem-solving behavior in the last 20 years has investigated several approaches based on the Newell and Simon (1972) information-processing framework. In the current perspective, understanding and solving a problem are built up from

The research was supported by the URA CNRS-1297.

We would like to thank three anonymous reviewers for helpful comments on an earlier version of the article.

Correspondence and requests for reprints should be sent to Jean-François Richard, Laboratoire de Psychologie, Cognitive du Traitement de l'Information Symbolique, Université de Paris 8, 2 rue de la Liberté, F 93526 St. Denis Cédex 02, France.

several components: the interpretation of the problem statement (mainly understanding which actions are allowed), the transfer of known procedures to similar problems (analogical transfer), the use of general heuristic rules, particularly goal-production rules, and the history of the problem, which can be defined as the memory of visited states.

A number of studies have presented probabilistic models to account for the behavior of a group of subjects. In these models, the estimated values of the descriptive parameters reflect both intrinsic properties of solving processes and differences among subjects. As mathematical models, they introduce probability values either for parameters of equation systems (Atwood & Polson, 1976; Jeffries, Polson, Razran, & Atwood, 1977), or for the simulation of probabilistic activation of rules in production systems. This has been done, for instance, in the modeling of problem-solving situations for a variety of river-crossing problems (Schmalhofer & Polson, 1986) and the simulation of the action side of rules when solving the Tower of Hanoi problem (Karat, 1982). These models afford a global description of solving processes that is of particular interest because it is an attempt to specify the underlying mechanisms involved in problem solving. However, these models do not capture fine-control and decision-making structures; they do not address the issue of how these underlying mechanisms might be organized at the individual level in order to be used for detailed line-by-line analysis of protocols.

In contrast to probabilistic approaches, computational models provide deterministic production systems in which the activation of rules depends on associated conditions. These are developed in order to model decision making at the individual level and are tested by line-by-line analysis of individual protocols. They are particularly suitable to simulating problem-solving strategies (Klahr & Robinson, 1981), learning by discovery, and shifting between strategies as well (Anzaï & Simon, 1979; Nguyen-Xuan & Grumbach, 1985). In these models, selection of rules that share the same set of conditions is made using general principles such as the comparison of each rule's specificity degree and/or the comparison of each rule's position in the list of rules (Anzaï & Simon, 1979; Klahr, 1978; Klahr & Robinson, 1981, VanLehn, 1991). In addition, two step deterministic production systems have sets of rules reflecting classes of cognitive processes (fact memorization, construction and test of hypotheses, goal generation, etc.) that are activated at a higher level by a class of control rules (Nguyen-Xuan & Grumbach, 1985).

However, although there may be very good modeling of information acquisition and processing, this is not the only aspect of learning. Another important aspect of problem solving is the elimination of conceptions that are not consistent with solution processes. This conceptual change is a central issue for the notion of problem restructuration, which was emphasized by Gestalt psychologists (Duncker, 1945). To date, no attempts have been made

to characterize this mode of learning through a model that describes the underlying mechanisms in detail.

The goal of this article is to present the constraint elimination model as a general model of *when* and *how* subjects, in the course of problem solving, change their conception of how the problem has to be solved. The assumptions of the model are that these changes of conception improve understanding of the problem statement and provide better planning as well. This is a new approach to problem solving, and focuses on the role of restrictive interpretations and how their effects are modeled with implicit constraints (restrictions of lawful moves). The proposed model is not contradictory with previous models but simply more general. The fact that rules or mechanisms from previous models can be translated into constraints demonstrates that the model presented here provides the same results. The reverse, however, is not true.

What gives the model potential psychological validity and what differentiates it from typical rule-based models is that this approach models both impasses and solving processes. Another major point is the integration of planning in the model without move anticipation. Dropping implicit constraints does not in itself provide the solution, given that both planning and constraint management are in fact involved in both insight and in plan-based problems. Planning is not presented here as depth of search but (a) integrated as constraints, and (b) dependent on the enlargement of the problem space that results from the dropping of constraints. Note that we provide no strict definition of a working-memory mechanism for storing and accessing a limited set of elements because memory content is represented in the form of dynamic constraints that may eventually be dropped. We do not develop the implications of the model for the study of contextual effects on short-term memory.

As a first attempt, the model was used to simulate the Tower of Hanoi solving processes because this problem is usually presented as a planning task whose difficulties involve planning consecutive and correct moves, not as a problem involving restructuring. The planning view is consistent with data on adults who are given a series of problems, but it is not consistent with data on children attempting to solve the problem for the first time. For instance, Byrnes and Spitz (1979) reported that the majority of children attempting to solve the Tower of Hanoi problem did not present a planning strategy on their first attempts before the age of 14. Although Klahr (1978) and Klahr and Robinson (1981) claimed that kindergarten children show planned behavior, the reason was that they used a very favorable context. We posit that the difficulty of the Hanoi problem is mainly due to misconceptions that are not consistent with the solution process. In this case, solving the Tower of Hanoi problem calls for the elimination of these misconceptions.

1. PROBLEM SOLVING AS ELIMINATION OF MISCONCEPTIONS

There is no doubt that misinterpreting the meaning of operators generates irrelevant conceptions of how the problem has to be solved. Misinterpreting can be either extensive or restrictive. Extensive misinterpretations lead to overshooting the explicit constraints about what may be done, which can be observed by the experimenter. On the other hand, restrictive interpretations produce "implicit constraints" in the sense that a subject will not do what can be done. Note that restrictive misinterpretations are not as easy to spot by an experimenter. Our position is that meaningful learning of problem solving primarily occurs through elimination of implicit constraints and, concomitantly, through successive novel conceptions of how the problem has to be solved that are incrementally more fully compatible with the solution process. For instance, subjects attempting to solve the well-known nine-dots problem draw lines from dots to dots rather than outside the dots. This restrictive interpretation of how to connect dots with a line generates constraints that reduce the problem space in such a way that the problem cannot be solved until another interpretation prompts the subject to draw the lines outside the dots.

Understanding from restrictive interpretation is not the only source of misconceptions. Irrelevant conceptions also occur through activation of inappropropriate procedures stemming from analogical transfer or the use of weak methods. There is ample evidence for inappropriate procedures in novice text editing (Waern, 1989), when weak problem-solving procedures are used to solve problems in new domains (Richard, 1990), or when general problem-solving heuristics, which are usually useful, are inappropriate for the current solution process. For instance, in attempts to solve a river-crossing problem, a means-end heuristic is inappropriate for moving two objects back to the starting bank, and thus, this is the most difficult move (Greeno, 1974; Simon & Reed, 1976; Thomas, 1974).

We claim that these various types of irrelevant conceptions have the same type of effect: They restrict possible moves and lead to specific states of the problem space where no further movement is allowed. The only possible action is then to revert back or abandon the problem. In an analysis of data on the Tower of Hanoi solving processes in 7-year-old children in an information-processing framework, Richard (1982) termed these specific states that lead subjects to impasses as dead-ends. This is a more accurate description of the effects of irrelevant conceptions than the observations of Gestalt psychologists on what they called fixation effects. However, as learning occurs, there are basic issues on how subjects change their understanding of the problem statement and the ways in which this competence should be modeled. The prime goal of this article is to address the psychological issue of how dead-end states, and the impasses that ensue, trigger conceptual changes when no recall mechanism for previous impasse solutions is available.

This aspect of problem solving cannot be easily modeled with classical production systems because we also need deterministic rules to explain why dead-ends happen, and rules to explain why they cease to be applied through learning. Consequently, classical production systems should have rules preventing the use of a sequence of operators, because this leads to an undesirable state (Anzaï & Simon, 1979). For instance, we need to be able to explain why and how subjects finally revert back, although they have the general heuristic rule: "Do not undo what has just been done." To do so, we need a formalism where rules and rule violations can be formulated easily. In contrast to classical production systems, an alternative approach consists in the elaboration of a formalism based on constraint satisfaction and on constraint elimination: a constraint being, in the narrow sense, the prohibition of specific actions.

In various fields (image understanding, schemata activation and filling in variable slots, design problems), the notion of constraint is acknowledged to be a basic notion in constraint satisfaction problems that are best-match problems (Hoc, 1990; Stefik, 1981a, 1981b). Thence, constraint satisfaction is used to provide the best solution in reducing the number of candidates. It permits the solving of large constraint satisfaction problems rapidly. For instance, it allows one to interpret a two-dimensional intensity image in terms of the depth and orientations of the three-dimensional surfaces in the world that give rise to that image or to "fill in" schemata in order to determine whether a particular candidate is an allowable assignment for a variable and, if the variable remains unfilled, to assign a default value. From this constraint satisfaction approach, a constraint is the set of possible combinations between values of associated variables, and common operations on constraints are their progressive refinement, their successive introduction into the problem, their propagation to trigger new constraint formulation, and their satisfaction that consists of producing the possible values of the associated variables.

In classical puzzle-like problem solving, up to now, the constraint approach has not been used to explain why the only candidate solution is not under the subject scope. To do so, the constraint elimination model may be used as the reverse approach of constraint satisfaction approach. In the former, constraints must be eliminated to reach the single state solution in a problem space; in the latter, constraints must be satisfied, or added, to choose among alternative solutions, which must at least satisfy a given number of constraints. We assume that puzzle-like problem solving, like single-match problems, requires mainly the elimination of constraints that prohibit specific actions.

There are two main advantages to a constraint elimination model for problem-solving simulation: (1) the various components involved in the cognitive processing of a problem are described in a common format, namely, as constraints; and (2) as constraints may not be of equal importance (some might

be absolutely imperative and others only desirable) and because in many cases it is not possible to satisfy every constraint, the least important ones may be disregarded. This is an important aspect of our modeling approach, and a new and important theoretical option, which consists of describing the problem representation a subject has, at a given time, as a set of ordered constraints. In the form of an ordered list, this set reflects the main component of what we call the *goal and constraints structure*. The current goal and constraints structure of the problem may differ across subjects. For a given subject, it emerges as a function of practice when the problem is repeated. Differences in solution processes then occur because some constraints are deleted and other are introduced.

In what follows, we first present the constraint elimination model and its implementation in a program. Second, we report the types of constraints we extracted from protocols of subjects solving the Tower of Hanoi problem with examples of constraints. Finally, we present the results of the simulation using the observed constraints.

2. THE CONSTRAINT ELIMINATION MODEL

Given the matrix of the entire set of possible moves and the entire set of states, the effect of a constraint is represented by a partition in two subsets of moves: moves consistent with the constraint and moves excluded by it. In this way, we are able to represent constraints that are expressed semantically in a negative way in the form of "IF (state or goal) DO NOT (actions)" as well as positive constraints, such as goal-related constraints, in the form of "IF (current goal) DO (actions)." Constraints, when defined in this way, are composed of (a) interpretations (or misinterpretations) of the instructions, (b) rules of choice related to goals or to sequences of goals (procedures), (c) general heuristics (i.e., do not make a backward move, do not make a memorized move), and (d) goal-related heuristics.

Each constraint is associated with its matrix in the form of a table that provides the constraint compatibility of each of the possible moves for each state (0 for compatible moves, 1 for incompatible moves). In order to make adaptive applications of constraints, the model assumes that some of the constraints are dynamic. In contrast to static constraints, where the matrix remains the same for the entire problem, dynamic constraints have a matrix the changes from trial to trial, for instance, with constraints such as "do not make a backward move," "do not repeat the preceding action." In addition, the strength of constraints is described as ordered positions of the constraints in a list, such that the least important ones may eventually be dropped from the list. More precisely, constraints are grouped into ordered classes. When each class consists of only one constraint, the set of constraints is completely

ordered. The model has three components: (1) the current goal and constraints structure; (2) the mechanisms producing the current goal and constraints structure; and (3) the decision and control mechanism.

The Current Goal and Constraints Structure. The current goal and constraints structure of the task, at a given time in the solving process, consists of a set of ordered constraints (as given later), a set of stored states and moves about the problem, and finally, a stack of goals, one of which is the current goal.

The Mechanisms Producing the Goal and Constraints Structure. The mechanisms producing the goal and constraints structure are modeled with goal production rules and memorization rules. The current goal in each state is given by the goal production rules in the form of general rules: "If some action has to be performed on an object, then do it on each of its parts." "If the goal has a prerequisite that is not satisfied, then take this prerequisite as the current goal." Goal productions rules are given specific contents in the context of a given problem.

Consider, for instance, a decomposition and ordering rule (DR) in the context of the Tower of Hanoi problem that consists of the transfer of a three-disk tower from one peg to an another:

DR: *If the goal is to move a 3-disk tower from Peg A to Peg C,*
Then first move the larger disk to Peg C, second the medium, third the small one.

According to this rule, [Disk 3 on Peg C] is the current goal for every state where Disk 3 is not on Peg C. Disk 2 on Peg C is the current goal for every state where Disk 3 is on Peg C, and Disk 2 is not on Peg C. Disk 1 to Peg C is the current goal for every state where Disk 2 and Disk 3 are on Peg C and Disk 1 is not on Peg C. Combining this rule with more powerful goal-production rules adds intermediary subgoals: For instance, Disk 2 on Peg B in states where Disk 3 is not on Peg C. The generation of the current goal for each state is also produced by a procedure. The rules for generating goals are in the same form of matrix or filter tables. This is possible as long as simple rules and simple problem spaces are considered.

Afterwards, given a current goal and a given state, a basic goal-related constraint and its matrix are defined: A goal allows every move that satisfies it (0 in the matrix) and prohibits all the others (1 in the matrix). For instance, in the context of the Tower of Hanoi problem, when the goal is in the form of (MOVE disk x on peg y), one move is allowed. When the goal is in the form of (REMOVE disk x), two moves are allowed. Finally, for each state, combining the current goal matrix with the basic goal-related matrix produces a final matrix that provides the set of prohibited moves related to goals. The program was given two memorization rules (MR).

MR 1: *If a move completes a goal,*
Then store it in memory as a pair of two states (preceding and resulting states).

MR 2: *If a state is such that, given the set of constraints, no move is allowed,*
Then store it in memory as a dead-end
and store the move that is done as a disregard of a constraint.

In addition, it may be useful to have a rule for the storage of the first move of the problem.

The Decision and Control Mechanism. In the constraint elimination model, the decision process consists of three steps. First, in Step 1, there is a check to ensure that the action is consistent with the constraints and can complete the running goal. If there is such an available action, it is executed. If there are concurrent available actions, one of them is chosen randomly. If the move is legal, the process is ended. The performed action is then stored in memory and the next encoded subgoal in working memory is taken as the current running subgoal. If there is no available move or if the move that is done is illegal, goal production rules are activated. If a new goal is produced, the process starts again with this new goal as the running goal. If no goal has been produced, Step 2 starts.

In Step 2, only constraints are considered to decide what to do. Because the search for completing the goal failed and search for new goals failed, it looks for what can be done. If one move (or more) is consistent with the constraints, this move (or one of these) is executed and, if legal, the action is not stored in memory and the process starts again at Step 1. If no move is allowed, then Step 3 is implemented for dead-end states.

Step 3 is characteristic of dead-ends. A dead-end might be an opportunity for deciding that the problem is unsolvable, but because the subject is encouraged by the experimenter to solve the problem, something has to be done and this requires violation of a constraint. Hence, the last constraint in the list is temporarily disregarded. To disregard a constraint means that the constraint is temporarily deleted. Afterwards, the process is restarted at Step 1. If the current goal becomes attainable, the action leading to it is undertaken. If not, the system will search for a possible move and, if found, will perform it. If there is no possible move or if the action that has been undertaken is illegal, and hence not accepted, the last constraint on the list is disregarded in turn. This process of temporarily deleting the last constraint on the list in turn is pursued until an action is rendered possible, and is ended when the undertaken move is accepted as legal. This move is then stored in memory. The disregarded constraint that enabled a possible move is fully deleted because that one move, which is accepted as legal, violates a constraint, but is an opportunity to learn that this constraint is not appropriate to the situation.

This mechanism explains why in some states a subject will make backward moves or return to the beginning (when allowed to by the experimenter); actually, "do not make a reverse move" is a constraint that is assumed to be at the end of the list. This mechanism also explains why illegal moves are attempted in states where backward moves or pauses are observed. In their analysis of Tower of Hanoi problem-solving protocols, Spitz, Webster, and Borys (1982) reported that "Only in the retarded group did the rule violations increase (from 18% to 33%) from the four-step to the seven-step problem.... These data suggest that the retarded subjects understood the rules but occasionally restored to violations when they could find no ready solution to the problem" (p. 926). We suggest that these states are in fact dead-ends where illegal moves appear because a constraint in the instructions has been disregarded. This is mainly observed in flat states of the Tower of Hanoi: Subjects try to insert the large disk under the medium one (from [3/1/2], they try [-/1/23]) or put the small disk beside the medium disk on the middle peg (from [3/2/1], they try [3/(2/1)/-]).

3. IMPLEMENTATION OF THE MODEL

A program was developed to simulate any given set of constraints. Given an ordered set of constraints (instructional constraints, general heuristics, heuristics related to goals and memorization rules), the program implements the constraint-processing mechanism defined before and uses it to evaluate consistency between the model's and subject's behavior.

Simulation of Effects of Constraints. The entire effect of a constraint on any state of the problem space is the subset of moves it allows to be performed and the subset of moves it does not allow. To simulate this effect, constraints are in the form of a "filter table" where rows stand for moves and columns for states. Each cell of the filter table of a constraint contains a Boolean value that indicates if the move is available at this state. Value "1" is assigned if the constraint applies (prohibited move), value "0" if it doesn't apply (allowed move). From the whole set of moves, the table is used to filter those that are permitted for a given constraint and a given state. By computing the matrix sum of filter tables for the entire set of constraints, the program constructs the overall table that is used to filter the set of available moves for each state. An example of the way filter tables are summed appears in Table 1.

Static Constraints. Static constraints are related to the structural and permanent properties of states in the problem space, regardless of the path leading to these states. This includes constraints generated by both correct and incorrect interpretations of the instructions. The program supplies each

TABLE 1
Example of Filter Tables for Two Constraints and the Way the Values Produced by the Constraints for Each State are Computed in a Global Filter Table.

		state 1	state2
table for constraint 1	move 1	1	0
	move2	0	1
		state 1	state2
table for constraint 2	move 1	1	0
	move2	0	0
		state 1	state2
global table	move 1	2	0
	move2	0	1

of the static constraints with a filter table. It indicates for each state which moves are prohibited under this constraint. Goal-related heuristics are treated as static constraints as long as it is possible to build a filter table. For instance, the Tower of Hanoi rule, "If the goal is to move the large disk, do not place another disk on it," is a static constraint because, with its table, we can filter available moves for any state where the large disk is free.

Dynamic Constraints. As noted earlier, dynamic constraints involve the memory storage of some of the previously visited states. Following the subject's path through the problem space, dynamic constraints are built by the program and updated on-line. Dynamic constraints represent a restricted short-term memory to avoid the choice of a forbidden move that has just failed, as well as to identify moves that lead back to previous states, and a middle-term memory to identify states that are known to be dead-ends.

Computing the Effects of Constraints. Before computing the set of available moves for the current state, the program updates each of the dynamic constraint filter tables, by deleting or storing forbidden moves in memory according to the meaning of the constraint. Note that the last table in the list is the goal-constraint matrix. It then adds updated dynamic filter tables to static filter tables to build the global filter table. Each value summarizes the number of constraints forbidding a given move in a given state. Note that cells in the global filter table contain integer values.

By looking at the global filter table, the program selects moves each time a zero value is found in the column of the current state. If one move is found, then the program enters the next state. This is Step 1 in the decision and control mechanism. If no move is found, the matrix relating to the basic goal-related constraint is dropped and the program looks to see whether there is at least one cell containing a zero in the new global filter table. This is Step 2, where the goal is left aside. Then it is checked to see whether there is a move consistent with the other constraints. When there is no move allowed in a state with regard to the constraints list rendering the problem unsolvable, we are in a dead-end. The program then discards the lowest constraint. To do so, the program considers the current state column of the lowest filter table and the relating column of the global filter table and subtracts the content of the former from the latter. Then the decision process starts again looking for at least one move. The constraint elimination process is continued until a move is found. Then, the list of constraints is restored for the next trial.

Testing the Model Through Simulation. The program provides a direct matching test of the list of constraints against the subject's protocol. The matching test is done through a main control loop that operates as follows: For each move in the subject's protocol, (a) program computes the set of available moves for the current state, that is, the set of moves from among those it would have randomly chosen in order to play; (b) the program is given the move that was performed by the subject; (c) if the subject's move is in the set of selected moves, the model is said to be consistent with the subject's behavior, otherwise, it is not; finally, (d) the program is given the move that was performed by the subject with the resulting state and the process starts again, so that the simulation always tracks the protocol.

4. IDENTIFICATION OF CONSTRAINTS FROM PROTOCOL ANALYSIS OF THE TOWER OF HANOI

Because constraints are sometimes mixed with others and because some are not expected, they are very difficult to detect. Thus, the identification of constraints requires careful analysis of individual protocols. This is probably the reason why, despite the large number of studies on the Tower of Hanoi problem, they have yet to be mentioned in the literature.

Recall that the purpose of this study is to investigate how the constraint elimination model can simulate the Tower of Hanoi problem solving with both impasses and solving processes, and that the model allows violations for constraints low in the list and then temporarily dropped. In order to constitute the goal and constraints structure, expected constraints need to be listed from the task analysis and from protocol analysis. To do so, we

first collected a full set of constraints in the Tower of Hanoi problem. Most high-level constraints, which are goal-related constraints, have already been described in the literature, especially in Anzaï and Simon (1979), Klahr and Robinson (1981), Spitz, Minsky, and Besselieu (1984).

We identified other constraints, mostly low-level constraints, from an extensive study involving 495 protocols of 7- to 8-year-old children solving the simple Tower of Hanoi problem three times in succession with three different size disks: a large disk (3), a medium disk (2), and a small disk (1), with the smallest on the top. The task was the classic three-disk problem consisting of the transfer of the three disks from the left peg to the right peg. The three pegs were arranged on a board from left to right (Peg A, Peg B, and Peg C): There was no rod and the disks could be piled directly on each other, so that it was possible to remove a disk that was not on the top. The problem statements were the standard ones: Move only one disk at a time, only move a disk that is at the top of a pile, don't put a disk on a smaller one.

From protocol analyses, we first collected data on critical events that were evidence of impasses, that is, states in which, in the subject's mind, all moves ahead are blocked. Critical events included: (a) long pauses of at least 30 s, prompting experimenter intervention to encourage the subject to do something; (b) backward moves; and (c) illegal moves. These critical events occur in the same states and often at the same time. These states are infrequent and clearly identified (Richard, 1982). Then, we identified what we call "episodes." Episodes are sequences of moves limited by the presence of one or more of these three critical events. A statistical analysis of these episodes revealed that the vast majority belonged to a rather limited number of types, about 30, and that there were symmetries among some types (Richard & Poitrenaud, 1988).

Prior to the model described here, a finer analysis (Richard, 1990) showed that it was possible to generate most of these episodes by means of a small number of rules. The basic idea in this earlier analysis was as follows: If a move that is allowed by the instructions is consistently avoided, that means that there is a rule prohibiting this move, then look for a reasonable rule that could prohibit this move. The analysis of the content of these rules, and subsequent experimental verifications, led us to group these rules into four subsets: interpretation of instructions, analogical transfer of procedures, meta-rules of action, and memory of visited states.

In this study, the procedure we used to identify the ordered list of constraints in a given protocol, consisted of determining whether each possible low-level constraint was consistent or not for each successive move in the protocol. Constraints with a substantially large number of violations were avoided. Constraints with few or no violations were retained and ranked as a function of the entire number of violations. This provided a basic ordered list that was used to simulate the protocols. In addition, local adjustments

were made to obtain finer adjustment: by small changes in ordering, permanent dropping of a constraint that had been frequently violated, the introduction of a goal-related constraint, or a constraint related to the instructions. (When attempting to perform a prohibited move, the subject was reminded of the appropriate part of the instructions.)

Identification of candidate constraints to simulate a protocol is very much like parameter estimation in mathematical models. Given the decision rules in the model, it consists of finding the ordered list of constraints providing the best adjustment. This has been done by hand until now, but we are currently devising a program to do this automatically. If the protocol is long enough (20 trials in a problem or several problems), there is no or only one solution.

The constraints include explicit constraints related to the interpretation of instructions, implicit constraints related to the restrictive interpretation of operators, constraints due to the analogical transfer of procedures, goal production rules, goal-related constraints, meta-rules of action, and constraints due to the memory of visited states.

Explicit Constraints Related to the Interpretation of Instructions (IIEC). In the Tower of Hanoi problem, instructions are stated as explicit constraints.

IIEC 1: *Do not move more than one disk at a time.*
IIEC 2: *Do not move a disk which is not at the top of a pile.*
IIEC 3: *Do not put a disk on a smaller one.*

However, some constraints need to be inferred.

IIEC 4: *Do not insert a disk inside a pile.*
IIEC 5: *It is not allowed to take a disk which is not at the top of a pile.*

These contraints cannot be seen when disks with holes are placed on a rod. In this situation, because the disk can only be placed on the disk on top of the pile, and because a disk that is not on top of the pile cannot be removed, these examples are examples of constraints that cannot be misunderstood.

Implicit Constraints Related to the Interpretation of Instructions (IIIC). Other constraints may be inferred, although they are not relevant to the problem statement. As they stem from restrictive interpretation of the operators, they are very difficult to detect. Striking examples, however, can be found in the interpretation of the action "to move." We detected three alternative constraints related to this interpretation.

IIIC 1: *Do not jump over a peg (empty or not).*
IIIC 2: *Do not jump over a peg with a disk.*
IIIC 3: *Do not jump over a peg with a disk that is smaller than the disk being moved.*

IIIC 1 is the most restrictive: It prohibits moving a disk from left to right or the reverse. IIIC 2 and IIIC 3 may be seen as exceptions to this restriction: "Do not jump over a peg except if it is empty" (IIIC 2) and "do not jump over a peg except if it is occupied by a larger disk" (IIIC 3).

Consider that Peg B has a larger disk than the disk being moved from left to right starting from Peg A to Peg C. IIIC 1 or IIIC 2 renders this move possible by a succession of two moves: First, a move from left to middle, and second, a move from middle to right. By contrast, consider that Peg B has a smaller disk than the disk being moved from left to right starting from Peg A to Peg C. As IIIC 3 does not allow jumping a peg having a smaller disk, it also does not allow jumping from Peg A to Peg C, or to move successively from Peg A to Peg B and from Peg B to Peg C because the first of the two moves (i.e., from Peg A to Peg B) would be a violation of the explicit instructions, "do not put a disk on a smaller one." Thus, IIIC 3 ("do not jump over a peg with a disk that is smaller than the disk being moved") is likely to be more resistent to deletion than IIIC 1 and IIIC 2 ("do not jump over an empty peg," and "do not jump over a peg with a disk," respectively). This is what we commonly observed. This is illustrated in the subject's protocol data presented in (Table 2).

This suggests that the action "to move an object" was interpreted by subjects as "to move the object in the two dimensions of the plane" and was not correctly interpreted as "to move the object in the three-dimensional space," as intended by the experimenter. It is not unreasonable to assume that 7- to 8-year-old children have a prototypical way of moving, that is, moving in the two-dimensional plane, as in walking. Such a misinterpretation of "moving an object" is impossible in the situation used by Klahr (1978) and Klahr and Robinson (1981) because they used inverted cans and introduced a story in which the cans were monkeys jumping from tree to tree. In this monkey situation, "moving" might have been interpreted as "moving in the third dimension" and not "moving in the two-dimensional plane."

Constraints Due to Analogical Transfer of Procedures (ATP). In half of the protocols, we identified stereotyped sequences of moves. These sequences consisted of moves from the initial state [321/-/-] to state [3/2/1], back to state [321/-/-], then to state [3/1/2], back again to state [321/-/-] and [3/2/1]. Similar sequences appeared between the state where the Tower is in the middle [-/321/-] and the two flat states, [1/3/2] and [2/3/1]. This may be attributed to the application of a procedure that would be appropriate if there were four pegs instead of three. Our contention is that these stereotyped sequences of moves are in fact generated by the application of a common procedure in the manipulation domain that consists of two subgoals and the application of a meta-rule.

TABLE 2

Example of simulation of a protocol. A state of the solving path is given with disk position. [12/-13] means that the little (1) and the medium (2) disks are on the left peg, the large disk (3) on the right peg and that the middle peg is empty (-). Moves in the subject's protocol and selected moves of the model are described with the disk to be moved and its destination (A, B, & C for the left, the middle and the right peg respectively), 3 to C meaning, for instance, that the large disk has been put on the left peg. Right hand columns indicate the working-memory content (running goal and stored states) and some comments.

	STATE	MOVE		COMMENTS	
		Subject	Model	Goals & Memorization	Disregarded Constraints
PROBLEM 1					
First trial					
0	[123/-/-]	1 to C	1 to C , 1 to B	goals: [tower to C] [3 to C]	
1	[23/-/1]	2 to B	2 to B		
2	[3/2/1]	1 to B	1 to B		
3	[3/12/-]	1 to C	1 to C	memory: (state 3) & (1 to C)	dead-end: violation of C9
4	[3/2/1]	1 to B	1 to B	memory: (state 4) & (1 to B)	dead-end: violation of C9
5	[3/12/-]	3 to C	3 to C	memory: (state 5) & (3 to C)	dead-end: violation of C8
6	[-/12/3]	1 to A	1 to A, 1 to C	goal: [2 to C]	
7	[1/2/3]	2 to C	2 to C		
8	[1/-/23]	1 to C	1 to C	goal: [1 to C]	
9	[-/-/123]	problem solved			
Second trial					
0	[123/-/-]	1 to C	1 to C , 1 to B	goals: [tower to C] [3 to C]	
1	[23/-/1]	2 to B	2 to B		
2	[3/2/1]	1 to B	1 to B		
3	[3/12/-]	3 to C	3 to C		
4	[-/12/3]	1 to A	1 to A, 1 to C	goal: [2 to C]	
5	[1/2/3]	2 to C	2 to C		
6	[1/-/23]	1 to C	1 to C	goal: [1 to C]	
7	[-/-/123]	problem solved			
PROBLEM 2					
0	[-/-/123]	1 to B	1 to A, 1to B	goals: [tower to A] [3 to A]	
1	[-/1/23]	2 to A	2 to A		
2	[2/1/3]	1 to A	1 to A		
3	[12/-/3]	1 to C	(1 to B) then (1 to A) then (1 to C)		dead end: (see text)
4	[2/-/13]	2 to B	2 to B		
5	[-/2/13]	1 to B	1 to B, 1 to A		
6	[-/12/3]	3 to A	3 to A		
7	[3/12/-]	1 to C	1 to A, 1 to C	goal: [2 to A]	
8	[3/2/1]	2 to A	2 to A		
9	[23/-/1]	1 to A	1 to A	goal: [1 to A]	
10	[123/-/-]	problem solved			

ATP 1: *Remove all the disks which are at the top of the disk to be moved and place them on an empty peg.*
ATP 2: *Move the large disk.*
ATP 3: *If there are several ways of completing a goal and if one has been tried and has failed, Then start again and try a new one (familiar meta-rule).*

In state [3/2/1], the first subgoal is completed but the second one cannot be completed, such that backward moves to the initial state are generated. Then, another way of removing the disk at the top is tried and fails in the same way. This generates a stereotyped and cyclic behavior which is stopped when the subject notices that every possible path of moves has been tried. This requires a memory of preceding moves. The common sequences produced by this procedure are accounted for by the following analogical transfer constraint (ATC).

ATC 1: *Only move a disk to an empty peg or to the top of a pile of all the larger disks.*

Due to the addition of this constraint to instructional constraints, no move is possible in states [3/2/1] and [3/1/2], except moving back to the preceding state.

Goal Production Rules (GPR). The most primitive rules we have identified follow.

GPR 1: *If the goal is to move a three-disk tower from Peg A to Peg C, Then first move the larger disk to Peg C, second the medium, third the small one.*

This is an instantiation of the general rule "if some action has to be performed on an object made of parts, do it on each of its parts" combined with the idea that the disks have to be placed on Peg C in decreasing size order.

GPR 2: *If the goal is to move a given disk to a given peg and if there is a smaller disk on that peg, Then remove this smaller disk.*

GPR 3: *If the goal is to move a disk that is not at the top of the stack, Then remove the upper disks.*

GPR 2 and GPR 3 are instantiations of the general rule: "if the goal has a prerequisite that is not satisfied, then take this prerequisite as the current goal." Finally, optimal strategies are represented as goal production rules such as.

GPR 4: *If the goal is to move a disk placed on a starting peg to a second given peg, Then remove and place all the set of upper disks on the disk to be moved at a third peg.*

The application of GPR 1 combined with recursive application of GPR 4 generates an optimal strategy for tower-ending problems.

Goal-Related Constraints (GRC). Goal-related constraints are general heuristic rules.

GRC 1: *If the goal is to move a disk to a given location*
and if this move is not prohibited by the instructions,
Then do not move it to another place until it becomes possible to reach that goal.

That means that if the goal disk move cannot be performed (let's say move Disk 3 to Peg C), another disk must be moved. This constraint might be seen as an efficient heuristic: Because it is not possible to move the large disk, it avoids wandering in other parts of the problem-space, for instance, in parts where the large disk is on the middle peg.

GRC 2: *If the goal is to move a disk,*
Then do not place a disk on it.

GRC 3: *If the goal is to move a disk to a given peg,*
Then do not move a smaller disk to this peg.

GRC 2 and GRC 3 are instantiations of the general rule that generates subgoals from prerequisites. These prerequisites are assumed to be present in the subject's representation. These constraints are similar to those that were proposed by Klahr and Robinson (1981) as rules in their models and to those that were described by Spitz et al. (1984). Besides the basic goal-related constraint presented before, another goal-related constraint was found in some protocols. In contrast to the basic goal-related constraints, it only prohibits moves on the current goal disk that do not satisfy the goal. It may be stated as a goal-related heuristic (GRH).

GRH 1: *If you have the goal of moving a disk to a given peg,*
Then do not move it elsewhere.

This constraint remains active when the basic goal-related constraint has been removed. Every subject is assumed to have the basic goal-related constraint, but only some of them present the GRH 1 constraint, which may be seen as a specific heuristic. When the current goal is not attainable, it prohibits any action on the current goal disk. This is equivalent to "wait until the goal is attainable."

Constraints Due to Meta-Rules of Action (AMC). As proposed in most models, we included the three following metarules, which are another kind of general heuristic rule.

AMC 1: *If a subgoal has been completed,*
Then do not make a move that destroys it.

AMC 2: *Do not make a backward move (e.g., a move leading back to the previous state).*

AMC 3: *Do not move the same disk two times in succession.*

Constraints Due to the Memory of Visited States (VSC). We only took one such constraint into consideration.

VSC 1: *If the present state is stored in memory as having been previously visited, Then do not repeat the same move.*

5. SIMULATION OF PROTOCOLS

5.1 Simulation of Children Solving the Tower of Hanoi Problem

To illustrate how the control and decision mechanism operates with a given set of constraints, it is useful to examine the simulation of the protocol of an 8-year-old who solved the problem of the transfer of three disks from the left peg to the right peg (Problem 1) twice and then solved the reverse problem, which consists of transferring the three disks from the right peg to the left peg (Problem 2) once. When starting to solve the problem, it is assumed that the subject has the first goal production rule,

GPR 1: *If the goal is to move a three-disk tower from Peg A to Peg C, Then first move the larger disk to Peg C, second the medium, third the small one.*

and the following list of constraints:

C1 : *Move one disk at a time.*
C2: *Move the disk at the top of a pile.*
C3: *Do not put a disk on a smaller one.*
C4: *If the present state has been stored in memory as a dead-end, Then do not make a move that is stored in memory because it was made in that state.*
C5: *If a subgoal has been completed, Then do not make a move that destroys this goal.*
C6: *If the goal is to move a disk to a given peg, Then do not move it to another peg.*
C7: *If the goal is to move a disk, Then do not place another disk on it.*
C8: *If there is a smaller disk on the middle peg, Then do not move a disk from the left peg to the right peg (or vice versa).*
C9: *Do not make a backward move.*
C10: *Do not move the same disk twice in succession.*

The observed and simulated protocols are presented in Table 2. At the start, the running goal is "move the tower to Peg C." As it cannot be accomplished, three subgoals are generated and stored in working memory in the following order: [3 to C], [2 to C], and [1 to C]. Hence [3 to C] is the running goal, but it cannot be attained. As no other subgoal is generated, the moves consistent with the constraints are listed. There are two moves: [1 to C] and [1 to B]. One of them has to be chosen at random. Because the subject made [1 to C], the same choice is attributed to the model in order to continue the simulation.

In State 1, because the goal cannot be reached, the system looks for a move. Move [2 to B] is the only move consistent with the constraints because [1 to B] violates C10, [1 to A] violates C9 and C10. In State 2, the goal cannot be reached and only one move is allowed by the constraints: [1 to B]. In State 3, the paradoxical backward move [1 to C], made instead of [3 to C], which would have reached the running goal, can be explained as follows. Because [3 to C] violates C8, because [1 to A] violates C7 and [1 to C] violates both C9 and C10, State 3 is a dead-end. The suppression of C10 does not allow any move, so C9 is disregarded in turn and allows [1 to C]. This move is stored in memory.

State 4, once again, is a dead-end because [2 to A] violates C7, [1 to A] violates both C7 and C10, and [1 to B] violates both C9 and C10. No move is allowed by removing C10, [1 to B] is allowed by removing C9. Thus, [1 to B] is done and stored in memory. State 5 is the same as State 3 but the behavior cannot be the same because [1 to C] has been stored in memory as having been made in State 3, so that [1 to C] violates C4. The next constraint (e.g., C8) has to be disregarded and [3 to C] is allowed. The end of the problem is obvious.

At the end of the problem, C8 is permanently removed. C9 and C10 become C8 and C9. There is no other change in the list of constraints between the first and second trial of Problem 1. The second trial of Problem 1 was solved with the minimum number of moves. Thus, no change should appear between the list of constraints for the second trial of Problem 1 and the list of constraints at the start of Problem 2. However, Problem 2 presents errors. The simulation has to explain the subject's behavior in both problems. The main result is that simulation is consistent with the observed protocol of Problem 2. There is only one discrepancy on Problem 2 that occurs in State 4 where the model also predicts [1 to C], but after [1 to B] and [1 to A], successively. Simulation of other protocols shows that the model and the given constraints predict a single move on about 61% of the movements and that protocols can be perfectly simulated movement by movement. For instance, the simulation of a very long protocol (44, 21, and 7 moves for two trials on Problem 1 and one trial on Problem 2, respectively) was highly consistent with the subject's moves except for only 1 of the 72 movements.

5.2 Simulation of Anzaï and Simon's (1979) Practice Tower of Hanoi Situations

To illustrate the ways in which an individual's representation evolves through practice as described in this model via constraints and goal production rules, consider the behavior of the subject studied by Anzaï and Simon (1979) and reanalyzed by VanLehn (1991). The behavior of the first episode (Statements S1 to S23)—characterized by Anzaï and Simon (1979) as selective search strategy—may be simulated by the N-Disk goal production rules and a list of constraints as follows.

GPR 1: *If the goal is to move a pile of disks to the goal peg,*
Then move each disk to that peg starting with the largest.

GPR 2: *If the current goal is to move a disk to its goal peg*
and if there are one or two disks on that peg,
Then move it or move them to the other peg starting with the larger.

C1, C2, and C3 are instructional constraints.

C4 : *If a move has been stored in memory in a given state,*
Then do not repeat this move in that state.

C5: *If the current subgoal is to move one disk to a given peg,*
Then do not move it to another one.

C6: *If the current goal is to move a disk,*
if that disk was free, and another disk was placed on it,
Then do not leave this disk in this location.

C7: *Do not make a backward move.*

C8: *Do not move the same disk twice in a row.*

C9: *If Disk 2 has just been moved,*
Then place Disk 1 on Disk 2.

C10: *If the current goal is to move a disk to a given peg,*
Then do not move a smaller disk to that peg.

The simulation of this part of the protocol is presented in Table 3. Notice that C10 applies before the move and prohibits the move and that C6 applies after the move and cancels the move: C6 means that the violation of the constraint of not blocking a disk that has to be moved is not anticipated but is recognized after the move has been made. It is expected that when it has been applied, C6 is transformed into C6′: "*If the current goal is to move a disk, do not place another disk on it.*" As shown in Table 3, the set of constraints and goal production rules is sufficient to generate the observed protocol: It is a description of a *selective search strategy*. On Trial 12, the subject quits. The predicted possible moves in this state are [1 to B] and [1 to C]: [1 to C] would be a violation of C6′ and [1 to B] would revert back to the same state as in Trial 8, which explains why the subject prefers to give up.

TABLE 3
Simulation of the First Phase in the Protocol of Anzaï and Simon's Subject (1979)

	STATE	MOVE		GOAL	DISREGARDED CONSTRAINTS
		Subject	Model		
1	[12345/-/-]	1 to B	1 to B	5 to C	
2	[2345/1/-]	2 to C	2 to C	5 to C	violation of C10
3	[345/1/2]	1 to C	1 to C	5 to C	violation of C10 because of C9
4	[345/_/12]	3 to B	3 to B	5 to C	
5	[45/3/12]	1 to A	1 to A	2 to B	
6	[145/3/2]	2 to B	2 to B	2 to B	
7	[145/23/_]	1 to B	1 to B	1 to B	
8	[45/123/-]	4 to C	4 to C	5 to C	violation of C10
9	[5/123/4]	1 to A	1 to A,1 to C, 4 to B		
10	[15/23/4]	2 to C	2 to C	4 to B	
11	[15/3/24]	2 to B	2 to B	4 to B	violation of C7 because of C6
12	[15/23/4]	leave	1 to B, 1 to C, 4 to B		violation of C6'

In the second episode (S26 to S74), when again in the initial state, Anzaï and Simon (1979) reported that the subject moves Disk 1 to C instead of to B and explains "since 1 is the only disk I can move, and last time I moved it to B, I'll put it on C this time... from A to C. (p. 138)" This is explained in the model by the application of constraint C4, since the first move in the preceding episode was stored in memory. This episode leads to a solution to the problem with a single hesitation in state [-/1234/5] between moving Disk 1 to C or to A. Finally the correct move, Disk 1 to A, is chosen. This episode is also characterized by more expressions of goals. Behavior in this episode is simulated by means of updating the goal production rules.

First, GPR 1 is transformed into GPR 1′ in order to be generalized to any peg because it does not apply to the goal peg of the disk.

GPR 1′: *If the current goal is to move a pile of disks to a given peg,*
Then move each disk to that peg starting with the largest.

Second, GPR 2 is generalized to goals involving any peg and becomes GPR 2′.

GPR 2′: *If the current goal is to move a disk to a given peg*
and if there are one or two disks on that peg or that disk,
Then move it or move them to the other peg starting with the larger.

Third, a new goal production rule (GPR 3) is added. It explains the subject's verbal reports about goals.

GPR 3: *If the goal is to move Disk 5 to Peg C,*
Then move Disk 4 to Peg B.

Rule GPR 1′ allows for producing the subgoal Disk 4 to C in state [-/1234/5] and the subgoal Disk 3 to C in state [-/123/45]. Rule GPR 2′

allows for correct performance when the subgoal disk is blocked by two disks. In cases where the subgoal disk is blocked by three disks, it produces a dead-end if the move of the third blocking disk is incorrect.

The next episode is characterized by an optimal solution without hesitation and implementation of the recursive rule: "In order to move Disk 5 to C, move Disk 4 to B; in order to move Disk 4 to B, move Disk 3 to C; in order to move Disk 3 to C, move Disk 2 to B; in order to move Disk 2 to B, move Disk 1 to C." This is formulated in the model by the following rule GRP 4.

GPR 4: If the goal is to move disk n to a given peg,
Then move disk (n − 1) to the other peg.

This rule combined with rule GRP 1′ generates each move without any need for heuristic constraints and produces the optimal solution. In the last episode, there is no change in behavior but the goal decomposition rule and the recursive GPR 4 are integrated into a single recursive rule, GPR 4′.

GPR 4′: *If the goal is to move a pile of n disks to a given peg,*
Then move the pile of n − 1 disks to the other peg,
move disk n to the target peg
move the pile of n − 1 disks to the target peg.

These rules are equivalent or identical to Anzaï and Simon's (1979) rules. For the second and third episode, the model reduces to a production rule model, similar to the model proposed by Anzaï and Simon and the reinterpretation proposed by VanLehn (1991). Heuristic constraints play no role because each move is generated by goal generation rules and, as there are no misinterpretations of instructions, there is no need to make the instructions explicit in the model.

This analysis shows that rule production models can be embedded in the general framework of the constraint elimination model and may be seen as a simplification of the model. Similarly, the whole set of rules inferred by VanLehn (1991) from subjects' verbalizations are easily mapped onto the set of goal generation rules or constraints we have presented here. First, VanLehn (1991, p. 11) listed five rules present throughout the protocol. Rule, 1, "The top-level goals are first to get Disk 5 to Peg C, then get Disk 4 to Peg C, then Disk 3, Disk 2, and finally Disk 1," is goal production rule GPR 1. Rule 2, "It is illegal to move the same disk on consecutive moves," is C8. Rule 3, "If there is only one legal action, then do it," is embedded in the decision and control mechanism. Rule 4, "If there are multiple legal actions, but one of them is to put Disk 1 on top of Disk 2, thus forming a two-high pyramid, then do that action" is equivalent to C9.

Second, VanLehn (1991) listed a set of present rules that were eliminated (Rules SSS, 1 blk, 2 blk). The rules forming the Anzaï and Simon (1979,

p. 134 selective search strategy (Rules SSS), "Do not move the same disk on consecutive moves" and "do not move the smallest disk back to one peg it was on just before it was moved to its current peg" are, respectively, C8 and C7. The rules for removing a disk (rules 1 blk) or two disks (2 blk) blocking the disk to be moved are embedded in goal production rule GPR 2.

Finally, VanLehn (1991, p. 11) listed three rules acquired during the protocol (Rules 4B, Dsk, Pyr). Rule 4B, "Before attempting any of the top-level goals, try to get Disk 4 to Peg B," is GPR 3. Rule Dsk, for disk-subgoaling strategy, "if the goal is to get a disk from one peg to another, and if there are some disks blocking the move, then get the largest blocking disk to the peg that is not involved in the move," is GPR 2'. Ruel Pyr, that is the pyramid-subgoaling strategy, "If one goal is to move a pyramid from a peg to another peg, then get the next smallest pyramid to one peg that is not involved in the move" is GPR 4'.

Some of the constraints (C4, C5, C6, C9, and C10) present in the model do not have equivalent rules in the VanLehn (1991) analysis. These constraints explain the generation of impasses, manifested by pauses, hesitations, and verbalizations about the difficulty of the task. VanLehn was right to insist upon the importance of such events. We suggest that impasses are generated by the constraints mentioned before. Unfortunately, these constraints are probably too implicit to appear in subjects' verbalizations.

6. COMPATIBILITY OF THE CONSTRAINT ELIMINATION MODEL WITH OTHER APPROACHES

The constraint elimination model is not contradictory with other models that have been proposed for Tower of Hanoi problems. As seen in Anzai and Simon (1979) and VanLehn (1991), much of these previous models can be reformulated within the constraint elimination model framework by simply providing a set of constraints and a set of goal production rules that lead to practically the same simulation. This discussion of how production rule models are embedded in the constraint elimination model concludes by turning to models in which this choice is based on the result of one or more anticipated moves, and models in which the choice of a move depends on the application of rules to the current state.

6.1. Reformulation of Models with Moves Anticipation

Move-anticipation models are characterized by depth of search. Depth levels are clearly related to the notion of working-memory capacity, which has received a great deal of attention in the cognitive development literature. Hence the appeal of interpretation of problem-solving behavior in terms of depth of search. However, there is no convincing evidence and the question

remains open. Moreover, a crucial prediction lacks confirmation: anticipating moves takes time, and the more planning ahead takes place, the longer the first trial response latency should be. If depth of search is related to performance, response latency should be correlated with performance. Actually, Spitz et al. (1984) found no correlation in a study devised for this purpose. Moreover there is no difference between normals and retardates. On the other hand, it seems reasonable to assume that depth of search, which is constant at a given level of development, will not depend on previously solved problems.

Move-anticipation models assume that a search is made through generation of sequences of moves, which are smaller or equal to the maximum length of moves, that can achieve the current goal or subgoal. Alternatively, because choice of a move depends exclusively on the current state of the problem and not on the state resulting from an anticipated move, the constraint elimination model does not define different levels of depth search. However, some heuristic constraints have the same effects as anticipation. For instance, the constraint, "Do not move a disk on the top of the disk that is the current subgoal," is equivalent to an anticipated sequence of moves on Length 2 in problem situations where the disk to be moved is a single disk on the target peg. Similarly, "Do not move a disk (that is not the subgoal disk) to the target peg," is equivalent to the anticipation of two moves if, when the target peg is empty, the disk to be moved is on top of the subgoal disk and has to be moved to the other peg.

Klahr and Robinson (1981) presented a set of models with move anticipation that differ in three features: rules for subgoal selection, competence to detect obstructors and remove them, and depth of search. Goal generation rules are similar to rules we have proposed for decomposition of goals into subgoals and subgoal ordering. Rules for obstructor detection and removal are included in GPR 2. Otherwise, they may be formulated as two goal production rules in the form of "If the subgoal disk is not free, then remove the disks on top of it (or move them to a given peg)," "If the subgoal solution peg is occupied by one or more smaller disks than the subgoal disk, then remove them (or move them to a given peg)." Differences between the constraint elimination model and an anticipation model such as Klahr and Robinson's is that constraints may be inadequate in some circumstances. Constraints produce errors or even dead-ends. This is not possible in Klahr and Robinson's model, where the goal may be attained within the anticipation range. Actually, Klahr and Robinson's model 8, which approximates the choices of the largest number of children, generates behavior that is very similar to an earlier version of the constraint elimination model, which was developed with rules for subgoal decomposition and subgoal ordering: a rule for removing obstructors on top of the subgoal disk, rules for goal production, instructional constraints, and two constraints C6 and C7, which we used to simulate children's protocols ("If there is a subgoal for moving a

disk to a given peg, then do not move it to another peg" and "Do not move a disk on top of the subgoal disk").

In a series of studies, Spitz and his co-workers (Borys, Spitz, & Dorans, 1982; Spitz et al., 1984) compared groups of retardates and normal controls on problems of two to seven moves, and analyzed their data within the framework of anticipation strategies similar to those presented by Klahr and Robinson (1981). Borys et al. described six strategies.

Strategy 1 is the optimal strategy. As previously shown, it can be easily expressed by a sophisticated goal generation rule. Strategy 2, as described by Borys et al. (1982), "checks whether it can move the largest disk not yet on the target peg to that peg, and if not, checks the source and target peg for possible obstructors. For the 7-moves problem, it notes that two disks, the small and the medium, are blocking the largest, and tries to find the minimum number of moves necessary to transfer the largest to the target peg" (p. 100). This strategy can anticipate four moves, including the move completing the subgoal and can solve the seven-moves tower-ending problem. Strategy 2 is achieved in the constraint elimination model through goal generation rules for subgoal decomposition and subgoal ordering, a rule for the removal of an obstructor on top of the subgoal disk (*If there is a disk on top of the subgoal disk, then remove it*) or on the target peg (*If there is a disk smaller than the subgoal disk on the target peg, then remove it*), the three constraints from the action meta-rules previously described, and three heuristic constraints: "If the goal is to move a disk, then do not place a disk on it," "Do not move the disk that is immediately larger than the subgoal disk to the target peg," "Move Disk 1 to the target peg." The third constraint, which is not a general constraint but appropriate for three-disk problems, can be learned in a context where, in a problem of transfer of three disks from the left peg to the right peg, the small disk is mistakenly moved to the middle peg. This will cause a dead-end because of the second constraint: The middle size disk is not allowed to move. If we assume the presence of a learning rule of the type "do not make a move that will lead to a dead-end," moving the small disk to the right peg will be avoided.

Strategy 3, described by Borys et al. (1982, pp. 100–101), is equivalent to the model described before except that the first and third heuristic constraints are dropped: "Do not move the disk which is just larger than the subgoal disk on the target peg" and "move Disk 1 to the target peg." Strategy 4 is formalized similarly by the Strategy 3 model with the addition of the following constraint: "If the goal is to move a disk on a given peg, then do not move a smaller disk to this peg." This always produces an error on the first trial of a three-disk problem, since Disk 1 is moved to the middle peg.

Note that this reformulation suggests that there may be transition from Strategy 4 to Strategy 3 through a learning process. The constraint, "if the goal is to move a disk on a given peg, then do not move a smaller disk to this

peg," has probably been learned with practice in solving two-disk problems which were given by the Borys et al. (1982, p. 90) as familiarization problems because this constraint is crucial to solve this type of problem. This constraint is transferred to three-disk problems, which produces Strategy 4. Thus, this constraint is deleted, producing Strategy 3 and is replaced by two constraints that take into account the size of the obstructor disk. This produces Strategy 2.

Strategy 5 is described by the same Strategy 4 model, except that the rule for the removal of an obstructor on the target peg is absent. Strategy 6 has been termed "a random strategy lacking means-end analysis. It wants the largest disk to go directly to the goal peg and will make moves randomly or violate game rules to get it there" (Borys et al., 1982, p. 101). This strategy is described by a model that consists of a single goal production rule, which is the rule of subgoal decomposition and subgoal ordering without heuristic constraints and a lack of instructional constraints, so that many violations are produced. Note that the authors reported a low percentage of violations for each category of subjects. These violations are interpreted in the constraint elimination model as the result of dead-ends: In this case, the lowest constraint in the list is a constraint given by the instructions.

Finally, note that models with move anticipation are able to account for global aspects of choice behavior with a reasonable degree of approximation, but do not seem able to simulate trial-to-trial problem-solving behavior or practice effects over a series of problems. Nor do they seem able to explain unusual behavior such as backward moves or unexpected violations of instructions.

6.2. Reformulation of Models without Moves Anticipation

The present approach also provides for reformulating production rule models in which there is no move anticipation. To do so, rules for goal production are transferred as such and rules that express heuristics are reformulated as constraints. Consider, for instance, the model proposed by Karat (1982, p. 543). Several productions are in fact embedded in the decision and control mechanism:

EX: 1 *If the problem is solved,*
Then stop processing.

EX: 2 *If the current goal is solved,*
Then mark subgoal disk as solved.

EX: 3 *If there is an approved move in working memory,*
Then execute it.

PR: 1 *If there is no current subgoal,*
Then create a subgoal.

PR: 2 *If the subgoal is directly solvable,*
Then solve subgoal.

EV: 1 *If legal move in working memory,*
Then approve move.

Other productions are expressed as either goal production rules or as heuristic constraints, consider:

PR:3 *If subgoal solution is only blocked by small disks,*
Then enter moves for small disks.

It is related to the main goal generation rule, which produces three subgoals that are encoded in working memory in the following order:

1. Move Disk 1 either to the subgoal peg (where the current subgoal disk is located) or to the target peg (where the current subgoal disk has to be moved).
2. Move Disk 2 to the peg that is not the subgoal peg or the target peg.
3. Move Disk 1 to the top of Disk 2.

Consider:

PR:4 *If last move was Disk 1,*
Then enter move other disk.

It is replaced by the heuristic C8 already used in the simulation of the Anzaï and Simon (1979) protocol: "Do not move the same disk twice in a row."

Consider productions PR:5 and PR:6:

PR:5 *If last move was Disk 2,*
Then enter no loop move with probability P_2.

PR:6 *If no constraint knowledge,*
Then enter means-end move with probability P_1.

They both contain probabilities. PR:5 is expressed as a goal production rule and PR:6 is equivalent to a heuristic constraint:

(PR:5) *If Disk 2 has just been moved,*
Then place Disk 1 on Disk 2.

and

(PR:6) *Do not move Disk 1 to another peg than either the subgoal peg or the target peg.*

In our reformulation, a proportion $\{P_1 \cdot P_2\}$ of subjects exhibited both the goal production rule corresponding to PR:5 and the heuristic constraint corresponding to PR:6. A proportion $\{P_2 \cdot (1 - P_1)\}$ of subjects has the former but not the latter. A proportion $\{P_1 \cdot (1 - P_1)\}$ has the latter but not the former. A proportion $\{(1 - P_1) \cdot (1 - P_2)\}$ has neither the former, nor the latter. In this way, probability values are only defined at the group level, if we work on group data. There is no probability left at the level of individual

data. So our reformulation of Karat's (1982) model allows a test at the individual level and, for that reason, the constraint elimination model is more informative.

Of course, the three instructional constraints need to be reported in the list of constraints in our reformulation. In Karat's (1982) model there is no possibility for violation of instructions. Moreover, there is no possibility for dead-ends: Memorization rules never apply, because they only apply in dead-ends to prohibit moves. As there is no possibility of a dead-end, the constraints do not need to be ordered. Karat's model can be seen as a very simple case of our model; without misinterpretation and without inadequate heuristics leading to parasitic constraints.

7. DISCUSSION

The contraint elimination model refers to a learning-by-doing conception of problem solving similar to the Anzaï and Simon (1979) perspective. As a general model, it may be applied to other puzzle-solving situations and extended to different kinds of problems. As the program does not take the semantics of the situation into account, it can be used in any case where one can supply a problem space and an ordered set of constraints. Indeed, as long as dynamic constraints (general heuristic and memorization rules) are not specific to a peculiar situation, the fact that they are "built-in" rules does not affect the generality of the program.

The move-generation mechanism may be transported without any changes, since it is quite general, as are the constraints that express heuristic rules. The only change will involve constraints related to interpretations of the instructions and those related to analogical transfer from known problems, which are highly situation-specific.

A good candidate problem for this is the missionaries–cannibals problem. Schmalhofer and Polson (1986) presented a production system that is close to our definition of constraints. Some of their rules can be directly integrated in the control and decision structure. For instance:

EV5 If a move clears the start bank
Then accept it

EX2 If a move is not detected as illegal
Then execute it (p. 117)

Other rules are similar to our heuristic rules:

EV3 Do not make a move which leads to the previous state

EV4 Do not make a move which leads to the start state

EV6 If a subgoal has been reached
Then do not make a move which destroys it (p. 118)

There is also a memorization rule, which is the same as ours, and other rules, which are in fact constraints on moves: "Do not make a move which leads to a state where the number of missionaries and cannibals are not the same," "Do not move less travellers than the maximum from the start bank," "Do not move more travellers than the minimum from the goal-bank" (p. 117).

The model proposed by Schmalhofer and Polson (1986) is a probabilistic model and, consequently, is tested using group data. The formulation of their hypotheses as constraints in the framework of the constraint elimination model would allow for the simulation of protocols to assist individual protocol analysis.

However, our study was primarily motivated by the observation that subjects usually solve a problem as a dual, but complementary task: planning while learning that they can do what they first supposed to be irrelevant, or not allowed in the task. This occurs even when solving puzzles that do not involve domain-specific knowledge. Thus, the main theoretical purpose of this article is to focus on the notion of restructuration that is modeled here as elimination of misconceptions. What is novel in the modeling of problem solving is to posit implicit constraints used to describe restrictive interpretations and to disregard implicit constraints used to change interpretations. We argue that this might provide a cognitive explanation for the elimination of misconceptions in learning by doing. Furthermore, given that successive elimination of the constraints generated by misconceptions enlarges the problem space with impacts on planning, the model integrates both plan-based and constraint-based approaches to problem-solving behavior.

8. CONCLUDING REMARKS

Our first remark concerns the need for models of misconception elimination dealing with what may be done in order to solve a problem. Subjects tend to have misconceptions about how to solve the problem because they do not have full understanding of the instructions. In this case, problem solving is mainly the learning by doing of what may be done.

Richard (1982) reported that restrictive interpretations of instructions are one of the main sources of misconceptions. Thence, there are limitations on generalizing models of problem solving based on the perfect understanding of instructions. In contrast to correct interpretation of explicit constraints, which delineates the task space, restrictive interpretations reduce the problem space by "implicit constraints." They can often make the problem unsolvable because subjects will not perform some of the actions that are necessary to attain the solution goal. For instance, when starting to solve the Tower of Hanoi problem, subjects frequently think that it is not legal to jump over a peg where a smaller disk than the disk being moved is located (IIIC 3). In order to solve the problem, they have to understand what may be done.

There are other sources of misconceptions of how to solve a problem. These include the transfer of inadequate procedures from previous problem-solving solutions, restrictions on operator scope that can be applied to some objects but not to others, restrictions of actions to prototypes of doing something, for instance, the well-known nine-dots problem that can be accounted for in this way. As shown by Richard (1989), restrictions on drawing lines to the edges of the square may be explained by a prototypic interpretation of the action of connecting two dots, which is drawing a straight line from one point, taken as the starting point, to the other, taken as the end point.

The constraint elimination model provides the same description for both the explicit constraints of the instructions and the implicit constraints that arise from restrictive interpretations of what may be done. By providing an explanation of when and how irrelevant implicit constraints are disregarded, this model formalizes how problem understanding is revised.

However, there are several limitations to the current version of the model. First, some constraints can be eliminated to find an allowed move, but at the end of the trial they are restored. The learning mechanism allowing for a change of order in the hierarchy or even complete elimination has not yet been formalized. A possible learning rule—when a constraint that has been dropped, thus allowing an action that proves to be legal—would be the permanent removal of this constraint, depending on how tightly this action were linked to a goal. Second, the elimination of a constraint is identified a posteriori: If a constraint is violated on a given trial, it is not reinserted in the list for that trial. The introduction of a relevant constraint is treated in the same way: The constraint is not introduced in the list as long as it presents inconsistencies with the moves done. It is introduced (usually after a violation) when it becomes consistent.

The second remark concerns the need for models that simultaneously integrate elimination of misconceptions about what may be done in order to solve a problem and planning solutions for this problem. There is a frequent dichotomy between plan-based and constraint-based problems, and this dichotomy needs to be revised. In our view, problem solving consists of two major processes: elimination of misconceptions (restructuring) and planning a solution. Typical planning problems involve identification of relevant constraints and require compromises among several perceived constraints. Both planning and constraint management processes are in fact involved in both types of problems. However, as a function of the problem and the subject's previous experience, one process may be prevalent but, as a rule, both processes contribute to the solution. A supposedly well-defined problem, such as the Tower of Hanoi problem, when solved by children, mostly involves elimination of misconceptions. A typical restructuration problem such as the nine-dots problem, as shown by Weisberg and Alba (1981), is

not easily solved when subjects are told that they have to extend the lines outside the square; there is also a planning process. In summary, the constraint elimination model is an attempt to elaborate a more unified view of problem solving because planning is modeled as a kind of constraint that must be confronted with other constraints.

The third remark concerns the broad implications of the constraint elimination model as a general problem-solving model. First, rule production models can be embedded in the general framework of the constraint elimination model and they may be seen as a simplification of the model. This means that the constraint elimination model is not contradictory with production systems that simulate problem-solving behavior. This is because, as shown in Anzai and Simon (1979), heuristic constraints play no role because each move is generated by goal generation rules and, as there are no misinterpretations of instructions, there is no need to render the instructional constraints explicit.

Second, the approach here is not limited to Tower of Hanoi problems. It can be easily extended to other move problems. We are reanalyzing data from the missionaries–cannibals problem within the framework of this model. The only changes needed are the introduction of specific constraints related to instructions and goals.

Third, up to now, we have only considered move problems, but the approach could be extended to symbolic problems such as crypto-arithmetic, or addition and subtraction problems, but it requires a formalized description of procedures such as those proposed by Poitrenaud, Richard, and Denhiere (1989) and Poitrenaud, Richard, Pichancourt, Tagrej, and Tijus (1990). In such problems, constraints due to knowledge of available procedures for a given problem should be added. As a matter of fact, there is some similarity between this approach and some aspects of repair theory proposed by Brown and VanLehn (1980) to explain the generation of bugs in subtraction. An impasse, as illustrated in this theory, is a situation caused by an impoverished procedure where no action is allowed. Repairs are actions borrowed from analogous situations or modifications in the focus of the operation proposed to resolve the impasse. They are filtered by criticisms, which are in the form "Do not...(action) in...(context). They play the same role as constraints in the constraint elimination model: The action performed is consistent with the constraints. Repairs may be seen either as constraints elimination (e.g., disregarding the order of operands) or as procedures arising from analogical transfer. A further comparison would require the incorporation into the theory of a description of procedures.

Fourth, as we have shown, some constraints are provided by rules of action that have been learned through solving previous problems and that are transferred through analogical transfer. If these rules are specific to the source problem, they may cause errors when solving new problems. This is

the case for the rule stating that another disk than the subgoal disk does not have to be moved to the target peg. In particular, the constraint elimination model approach has the potential to capture the role of learning in the course of problem solving and the dependence of solving behavior on the type of previously solved problems.

One final remark should be made about the limitations of the model. Beside the hierarchy of constraints rules, a number of mechanisms are assumed to occur: Mechanisms underlying statement interpretation, transfer of known solutions, goal production, memorization, and decision. Although these are constituent mechanisms of a complete problem-solving model, we have only looked at the modeling of production of goals, memorization, and decision mechanisms. We assumed that the decision mechanism operates on an ordered list of constraints and that deletion and insertion of constraints occur in this set of constraint rules. However, because of the model's assumption that complementary constraints emerge from interpreting the problem statement and transferring known procedures, research needs to be developed from a complex theory of understanding and from a theory of procedural knowledge representation.

REFERENCES

Anzai, Y., & Simon, H.A. (1979). The theory of learning by doing. *Psychological Review, 86,* 124–140.

Atwood, M.E., & Polson, P.G. (1976). A process model for water jug problems. *Cognitive Psychology, 8,* 191–216.

Borys, S.V., Spitz, H.H., & Dorans, B.A. (1982). Tower of Hanoi performance of retarded yound adults and nonretarded children as a function of solution length and goal state. *Journal of Experimental Child Psychology, 33,* 87–110.

Brown, J.S., & VanLehn, K. (1980). Repair theory: A generative theory of bugs in procedural skills. *Cognitive Science, 4,* 379, 426.

Byrnes, M.M., & Spitz, H.H. (1979). Developmental progression of performance on the Tower of Hanoi problem. *Bulletin of the Psychonomic Society, 14,* 379–381.

Duncker, K. (1945). On problem solving. *Psychological Monographs,* 58(Whole No. 270).

Greeno, J.G. (1974). Hobbits and orcs: Acquisition of a sequential concept. *Cognitive Psychology, 6,* 270–294.

Hoc, J.M. (1990). *Cognitive psychology of planning.* London: Academic.

Jeffries, R., Polson, P.G., Razran, L., & Atwood, M.E. (1977). A process model for missionaries–cannibals and other river-crossing problems. *Cognitive Psychology, 9,* 412–440.

Karat, J. (1982). A model of problem solving with incomplete constraint knowledge. *Cognitive Psychology, 14,* 538–559.

Klahr, D. (1978). Goal formation, planning and learning by preschool solvers, or "My socks are in the dryer." In R.S. Siegler (Ed.), *Children's thinking: What develops.* Hillsdale, NJ: Erlbaum.

Klahr, D., & Robinson, M. (1981). Formal assessment of problem-solving and planning processes in preschool children. *Cognitive Psychology, 13,* 113–148.

Newell, A., & Simon, H.A. (1985). *Human problem solving.* Englewood Cliffs, NJ: Prentice Hall.

Nguyen-Xuan, A., & Grumbach, A. (1985). A model of learning by solving problems with elementary reasoning abilities. In G. d'Ydevalle (Ed.), *Cognition, Information Processing and Motivation.* Amsterdam: North Holland.

Poitrenaud, S., Richard, J.F., & Denhiere, G. (1989, June). *La description des procédures: une approche "orientée objet"* [The description of procedures: An object-oriented approach]. Paper presented at Colloque International Informatique Cognitive des Organisations ICO 89, Montreal, Québec, Canada.

Poitrenaud, S., Richard, J.F., Pichancourt, I., Tagrej, M., & Tijus, C. (1990, March). *La desription des procédures: Leur décomposition hiérarchique et leur role dans la catégorisation des objets.* Paper presented at Colloque de l'Association pour la Recherche Cognitive, Paris.

Polson, P., & Jeffries, R. (1982). Problem solving as search and understanding. In R.J. Sternberg (Ed.), *Advances in the psychology of human intelligence.* Hillsdale, NJ: Erlbaum.

Richard, J.F. (1982). Planification et organisation des actions dans la résolution du problème de la tour de Hanoi par des enfants de 7 ans [Planning and structuring problem-solving methods in the Tower of Hanoi problem by 7-year-olds]. *Annéee Psychologique, 82,* 307–336.

Richard, J.F. (1990). *Les activités mentales: comprendre, raisonner, trouver des solutions* [Mental processes: Understanding, reasoning, finding solutions]. Paris: Colin.

Richard, J.F. (1989). Interpretation biases in problem solving. In G. Tiberghien (Ed.), *Advances in Cognitive Science, 2.* Chichester: Herwood.

Richard, J.F., & Poitrenaud, S. (1988). Problématique de l'analyse des protocoles individuels d'observations comportementales [Problematic analysis of individual protocols in behavioral observations]. In J.P. Caverni, C. Bastien, P. Mendelsohn, & G. Tiberghien (Eds.), *Psychologie cognitive: Concepts et méthodes.* Grenoble: PUG.

Schmalhofer, F., & Polson, P.G. (1986). A production system for human problem solving. *Psychological Research, 48,* 113–122.

Simon, H.A., & Reed, S.K. (1976). Modeling strategy shifts in a problem-solving task. *Cognitive Psychology, 8,* 86–97.

Spitz, H.H., Minsky, S.K., & Besselieu, C.L. (1984). Subgoal length versus full solution length in predicting Tower of Hanoi problem-solving performance. *Bulletin of the Psychonomic Society, 22*(4), 301–304.

Spitz, H.H., Webster, N.A., & Borys, S.V. (1982). Further studies of the Tower of Hanoi problem-solving: Performance of retarded young adults and nonretarded children. *Developmental Psychology, 18,* 922–930.

Stefik, M. (1981a). Planning with constraints (MOLGEN: Part 1). *Artificial Intelligence, 16,* 111–140.

Stefik, M. (1981b). Planning and meta-planning (MOLGEN: Part 2). *Artificial Intelligence, 16,* 141–170.

Thomas, J.C. (1974). An analysis of behavior in the hobbits-orcs problem. *Cognitive Psychology, 6,* 257–269.

VanLehn, K. (1991). Rule acquisition events in the discovery of problem-solving strategies. *Cognitive Science, 15,* 1–47.

Waern, Y. (1989). *Cognitive aspects of computer-supported tasks.* Chichester, England: Wiley.

Weisberg, R.W., & Alba, J.W. (1981). An examination of the alleged role of fixation in the solution of several insight problems. *Journal of Experimental Psychology: General, 110*(2), 169–192.