Second Annual Conference on Advances in Cognitive Systems Poster Collection (2013) 111-122

Elaborations on a Theory of Human Problem Solving

Pat Langley PATRICK.W.LANGLEY@GMAIL.COM
Nishant Trivedi NHTRIVED@ASU.EDU
Institute for the Study of Learning and Expertise, 2164 Staunton Court, Palo Alto, CA 94306 USA

Abstract

In this paper, we present an extended account of human problem solving and describe its implementation within ICARUS, a theory of the cognitive architecture. We begin by reviewing the standard theory of problem solving, along with how previous versions of ICARUS have incorporated and expanded on it. Next we propose four additional elaborations that bring the framework into closer alignment with human problem-solving abilities. After this, we report results on a number of domains that demonstrate the benefits of these extensions. In closing, we discuss related work and note promising directions for additional research.

1. Introduction

The ability to solve novel problems is one of the hallmarks of human cognition, and the study of this phenomena played a central role in the founding of artificial intelligence. Early models of problem solving provided some of earliest evidence for the computational nature of human thought. These accounts remain some of the most precise and detailed in the literature on mental phenomena, and the topic remains a central and important one that deserves continued study by scientists in the cognitive systems community.

Nevertheless, recent years have seen little AI research on problem solving that connects to psychological findings, and there has been minimal work on the topic in cognitive science, presumably because many believe our current level of understanding is sufficient. In contrast, we hold that current accounts of human problem solving, although generally accurate, remain incomplete and that we need to develop more comprehensive models that provide even deeper insights into this intriguing and complex aspect of cognition.

In this paper we report progress toward developing a theory of problem solving of this sort. We begin by reviewing the main tenets of the standard theory and some elaborations reported in our earlier work. Next we review ICARUS, a theory of the human cognitive architecture, focusing attention on how its problem-solving module incorporates these theoretical ideas. After this, we note some limitations of the existing framework and describe four extensions to ICARUS that attempt to address them. We close the paper by reviewing related work on problem solving and outlining directions for additional research.

2. The Standard and Extended Theories

In a previous paper (Langley & Rogers, 2005), we reviewed the standard theory of problem solving, first proposed by Newell, Shaw, and Simon (1958), which characterizes in computational terms how people respond to unfamiliar tasks. Early research in their framework dealt with abstract problems like the Tower of Hanoi and logical theorems, with later work shifting to semantically rich domains like physics and thermodynamics.

The standard theory of problem solving makes a number of increasingly specific claims about human cognition that are worth reviewing here. In short, these are:

- High-level cognition depends on the representation, interpretation, and manipulation of *symbol structures*. Newell and Simon (1976) later refined this idea into their *physical symbol system* hypothesis.
- Problem solving involves *search* through a problem space of candidate states that are encoded as symbol structures, that are generated from other states by applying operators, and that are tested through pattern-matching methods.
- Rather than carrying out exhaustive search through problem spaces, humans use *heuristics* to guide selective search; Newell and Simon combined this and the above claim into the *heuristic search* hypothesis.
- Problem solvers often utilize *means-ends analysis* to organize their search. This process finds differences between current and desired states, selects operators that reduce the differences, and applies the operators or creates subtasks to make them applicable.

Empirical studies have repeatedly found evidence that is consistent with this account of human problem solving, making it one of the most robust and stable contributions to our understanding of high-level cognition.

In our earlier paper, we proposed extensions to the standard theory that, we claimed, offered a more complete account of human problem solving. These included:

- Problem solving occurs in a physical context, with problem states corresponding to configurations of objects and operators mapping onto manipulations.
- The problem-solving process abstracts away from physical details during heuristic search, yet it returns to these details when implementing any solution found in this manner.
- Problem solving is seldom a purely mental activity; it typically interleaves internal reasoning about problem states with execution in the environment.

In addition, we reported an implementation of these and other extensions in ICARUS, a theory of the human cognitive architecture, and we used the system's behavior on a familiar task – the Tower of Hanoi – to illustrate and clarify their operation.

In the next section, we briefly review the ICARUS architecture and its approach to problem solving. Despite incorporation of the ideas noted above, our experience with its problem-solving module suggests that it lacks both the flexibility and the focused style of search that humans exhibit. This has led us to introduce further elaborations of the theoretical framework that we believe address these limitations, which we discuss later in the paper, along with their implementations.

3. ICARUS' Account of Problem Solving

ICARUS is a computational account of the human cognitive architecture that incorporates the extended theory of problem solving just summarized. Although we have described the framework in detail elsewhere (Langley, Choi, & Rogers, 2009), we should review its main tenets here as background for the extensions we present later. The architecture includes four modules – responsible for conceptual inference, skill execution, problem solving, and skill acquisition – that we should present in that order, since each builds upon its predecessor.

Conceptual inference is a mechansism for generating beliefs that describe the agent's current situation in terms of known concepts. On each cycle, the environment deposits percepts – attribute-value descriptions of perceived objects – into a perceptual buffer. The inference process compares these against conceptual rules, similar to Horn clauses, using matched cases to infer beliefs – instances of conceptual predicates – to memory. More abstract conceptual rules match against these inferences to produce higher-level beliefs, and so on. This bottom-up processing continues until it computes the deductive closure of the agent's conceptual rules and its percepts.

Skill execution is responsible for carrying out complex activities over time. This mechanism makes use of the beliefs produced by conceptual inference, but it operates in a top-down manner driven by the agent's goals. On each cycle, ICARUS finds the highest priority goal that is not satisfied by current beliefs. The system then selects a skill, similar in form to a hierarchical STRIPS operator, that would achieve this goal as one of its effects and that has conditions which match current beliefs. If the skill is primitive, the architecture carries out the associated actions to alter the environment. If the skill is nonprimitive, then the system selects the first of its subgoals not yet true, retrieves a skill with matched conditions that should achieve it, and either applies it (if primitive) or chains further (if not). ICARUS takes only one step through the skill hierarchy on each cycle, checking for new inferences before making additional ones.

Problem solving occurs when ICARUS' skill-execution process cannot find an applicable path through the skill hierarchy that is relevant to its current goal. In such cases, it calls on a form of means-ends analysis (Newell, Shaw, & Simon, 1958) that attempts to compose known skills in a dynamic fashion and to execute them as they become applicable. The first step involves selecting a skill with the goal as one of its desired effects or, if none exists, selecting a conceptual rule with the goal in its head. In the former case, it determines which of the skill's conditions are unsatisfied, selects one of them, and generates a subgoal to satisfy it. This leads the problem solver to select another skill that would achieve the subgoal.

This process continues to chain backward recursively off subgoals until it retrieves a relevant skill with satisfied conditions, in which case ICARUS calls on the execution module to carry it out in the environment, or until it fails, in which case it backtracks. Chaining off a conceptual rule also leads the system to create a subgoal, but not directly to skill execution. Upon achieving the current subgoal, the architecture turns its attention to other unmatched conditions of the selected skill or concept, continuing until it has satisfied them all. Such a situation lets it execute the selected skill or make inferences with the selected concept, which alters the belief state and lets it address any remaining issues. This activity continues until, if successful, the system achieves the top-level goal that initiated problem solving.

This process lets ICARUS address situations in which it lacks skills to achieve its goals, but it often requires the system to carry out substantial search. For this reason, the architecture includes a *learning* module that generalizes problem solutions and stores them as new skills that apply in similar situations. This operates whenever the problem solver achieves a goal or subgoal, which the system uses to index the new skill. The structure includes subgoals that problem solving satisfied along the way, along with conditions that were present when it began working on the task. After learning, ICARUS retrieves these new skills in similar situations so that it achieves goals by automatized execution rather than deliberative problem solving.

4. Extensions to ICARUS' Problem Solving

Although ICARUS incorporates the main claims of the standard theory and some important extensions, our experience with the architecture suggests that its account of problem solving remains incomplete. Difficulties we have encountered range from awkward programming conventions to implausible amounts of search to basic inability to reason about certain types of problems. Analysis of these drawbacks has led to a new implementation of ICARUS' problem-solving module that reflects four new claims about the nature of this activity. These do not contradict any of the earlier tenets, but they do elaborate the theory along interesting dimensions. In this section, we present each extension in turn, including our reasons for introducing it, the representational implications, and its import for processing.

4.1 Problem-Based Organization

The basic ICARUS problem solver is driven by individual goals. The system begins with a single top-level goal that leads it to generate a subgoal based on the unsatisfied conditions of a skill or concept. To present the architecture with tasks that involve achieving a conjunction of goals (such as occur in the Tower of Hanoi), the modeler must define a concept for that conjunction and refer to it in a single goal. This requirement is more than simply awkward or inelegant: it places limits on high-level domain skills, whether manually crafted or learned. This one-goal focus also keeps the system from reasoning about interactions among different goals.

In contrast, we hypothesize that humans organize their problem solving around *problems* and their subproblems. We have incorporated this assumption into a new version of ICARUS that operates on problems stated as sets of goals that may share variables. ICARUS utilizes this more comprehensive structure not only for top-level tasks but also for any subproblems that it generates during problem solving. Protocol studies make it clear that humans usually focus on one goal at a time, but also that they do this in the context of their sibling goals.

This new organizational commitment has important implications for the process of problem solving. Rather than checking to see whether it has achieved a goal, the new module checks to see whether it has satisfied the combination of goals that define the current problem. Moreover, chaining backward off the conditions of inapplicable skills or concepts leads it to create entire subproblems rather than individual subgoals. ICARUS retains the ability to focus on a specific subgoal, as this seems useful for selecting skills or concepts off which to chain, but they play a subordinate role to problems, which serve as the new architecture's main organizing principle.

4.2 Alternative Problem Formulations

The original ICARUS problem solver selects a subgoal by examining the conditions of the skill or concept instance it is attempting to satisfy and then picking one that does not match the current belief state. Unfortunately, this haphazard choice ignores information about the structure of the problem space that can prove important if the selection does not work out. The introduction of problems and subproblems as explicit structures in memory clarifies the situation, as it reveals that the current problem state may partially satisfy the given set of conjunctive goals in more than one way. This suggests the need for an extra level of decision making that involves choosing among different *formulations* of the current problem.[1]

We posit that the selection of such problem formulations plays a central role in human problem solving. We have incorporated this idea into the new version of ICARUS, casting the choice in terms of different *bindings* for variables in the problem. An example from the blocks world should clarify the notion. Suppose the current problem involves achieving three conjunctive goals with shared variables – *{(on ?x ?y) (on ?y ?z) (on ?z ?w)}* – and the current problem state includes the two beliefs *(on A B)* and *(on B C)*. Selecting the bindings *{?x* $\rightarrow$ *A, ?y* $\rightarrow$ *B, ?z* $\rightarrow$ *C}* leads to the single unsatisfied goal *(on C ?w)*, while selecting *{?y* $\rightarrow$ *A, ?z* $\rightarrow$ *B, ?w* $\rightarrow$ *C}* produces a different goal, *(on ?x A)*. These constitute distinct ways of viewing the task that have different implications for selecting a skill to achieve them.

Besides introducing additional structures that ICARUS must store during search, this change also requires an extra level of processing. Whenever the system encounters a new problem or creates a subproblem, its first step is to generate a set of maximal partial matches between the problem and the belief state, and then to select one that will drive behavior. Because the architecture represents this choice explicitly, it can backtrack over it should a given alternative not lead to a solution. Although adding to the apparent complexity of problem solving, this change lets ICARUS solve classes of problems it could not handle before.

4.3 Chaining Over Negated Goals

Another limitation of ICARUS' original problem solver was its inability to reason over negated skill conditions with any generality. The system could create a subgoal based on a negated condition, but it could achieve it only if had another skill that produced the negation directly. However, there exist cases in which no such skill is known and one must instead chain over the negation of a defined concept. Because negations involve universal quantification, one need only satisfy one of the negated subconcepts to achieve the parent goal, but this requires special machinery.

We have incorporated such a mechanism into the new problem-solving module. When chaining over the negation of a defined concept *C* with the conjunctive body $A_1, A_2, \ldots, A_n$, it uses De Morgan's law to creates a new subproblem that comprises a set of *disjunctive* goals – $\{ \neg A_1, \neg A_2, \ldots, \neg A_n\}$ – rather than the standard conjunctive ones. The system labels the structure as disjunctive,

1. We use this term in a weaker sense than adopted in early work on problem reformulation (e.g., Riddle, 1991), which dealt with altering task representations.

so that it recognizes it as achieved whenever any member of the goal set becomes satisfied. Otherwise, the problem-solving mechanisms remain oblivious to the type of problem, letting it interleave conjunctive and disjunctive tasks as necessary.

4.4 Heuristics for Skill Selection

Our experience with the ICARUS problem solver suggests that it carries out far more search, in a much less directed manner, than humans on the same tasks. The introduction of problems and problem formulations as explicit structures does not help on this front, since if anything they increase the number of choice points. This situation is not surprising. Computational approaches to problem solving and planning have long been known to lack direction, but publications have conveniently downplayed this fact. Domain-specific knowledge can reduce search considerably, but this cannot be the only source of humans' advantage, as it seems to hold even for novice behavior on novel tasks. This indicates a need for more informed guidance of skill selection during problem solving.

We hypothesize that domain-independent heuristics determine the selection of skill (operator) instances used to achieve a chosen goal. The new problem-solving module incorporates two guides of this sort. The first heuristic prefers skill instances that achieve more of the unsatisfied goals in the current problem formulation. Intuitively, this bias should reduce the steps required to solve the task. The second heuristic prefers skill instances that have fewer unsatisfied conditions. Intuitively, this bias should reduce the number of steps needed to make the selected operator applicable and thus achieve its intended goal.

These two biases substantially alter the search that ICARUS carries out during problem solving. Rather than selecting among skill instances at random, the new architecture carries out much more selective exploration of the problem space. As we report shortly, the two heuristics offer different degrees of assistance in different domains, but together they cut down the amount of search substantially, bringing them closer to human levels. At the same time, they can occasionally mislead the problem solver down false paths, another phenomenon that is sometimes observed in human problem solving.

4.5 Interactions Among the Extensions

We should clarify that these four extensions to the ICARUS problem solver are not entirely independent. The commitment to problems as the organizing structure serves as a prerequisite for the others, in that it provides the content over which reasoning about problem formulations, negated goals, and domain-independent heuristics operate. Without the presence of problems as explicit structures in memory, the latter three mechanisms would not be possible.

However, as we have noted, the ability to consider alternative problem formulations and to chain over negated goals expands the problem space that ICARUS searches considerably. The new heuristics provide general, domain-independent ways to guide this search and make it more tractable, thus mitigating the increased number of choices. Like other heuristics, they offer no absolute guarantees about reducing the amount of search that is carried out, but in practice we have found they are often highly effective along this front.

5. Experiences with the Extended Module

To study the new problem-solving module's behavior, we ran it on a number of domains that we believed would exercise its capabilities. In this section, we summarize our experiences in three testbeds, in each case describing the types of tasks that arise, the knowledge provided to ICARUS, and the system's operation.[2] In closing, we also mention some additional domains on which we have tested the module.

5.1 Problem Solving in the Depots Domain

One of our studies involved Depots, a domain that involves number of locations with stacks of crates that one must move to a different location. Operators include using hoists to lift and drop crates and trucks to transport them. Achieving a goal typically requires loading a crate onto a truck, driving it to a target location, and stacking it the desired way. Some tasks also require moving other crates in order to make this possible. For this domain, we gave ICARUS 12 conceptual rules that let it recognize arrangements of crates, hoist location and status, and truck location and contents. We also provided four primitive skills for hoisting and dropping a crate, loading or unloading a crate into a truck, and moving the truck.

All goals in the top-level problem are grounded, so the problem solver has no choice of bindings. However, there are many possible focus goals, from which it picks at random. Each choice leads in turn to selecting a skill for dropping the relevant crate, which produces a subproblem based on its conditions, typically only one of which is unsatisfied – that some hoist be holding the crate at the desired location. Selecting a skill to achieve this end offers two options, whether to hoist the crate or unload it from a truck. The 'fewer unsatisfied conditions' heuristic leads ICARUS to select the hoist skill, but additional thought reveals this requires the crate to be at the desired location, so this loop causes it to backtrack and select the unload option. Continued reasoning along these lines eventually leads to executable skills and successful placement of the crate. The problem solver then repeats this process for every crate, halting after it has transferred each object to its desired location.

We also carried out a lesion study to examine the benefits of the two heuristics. We found that, in this domain, the heuristic for preferring skills with fewer unsatisfied conditions led to far greater reduction in search and backtracking than the maximum goals heuristic. In fact, the latter provided little advantage over runs in which the problem solver used neither guide. We hypothesize that this occurred because primitive skills have only a few effects and hence score the same on this criterion, leading it to provide little guidance.

5.2 Problem Solving in the Gripper Domain

Another study focused on the Gripper domain, which involves several rooms, a number of balls, and a multi-gripper robot that lets it hold one ball per gripper. The goal is to move particular balls to specified rooms, making as few trips as possible by carrying multiple balls at a time. For this

2. We should note that, although all of our runs involve fully grounded goals at the top level, the system often created subproblems with unbound variables that confronted it with choices about bindings and formulation.

domain, we provided ICARUS with ten concepts for recognizing relevant situations, such as whether a gripper is holding a ball and whether the agent is holding as many balls as possible. We also gave the system four skills, one of which was not only hierarchical but recursive.

As before, a problem specifies desired locations for each ball, so the problem solver need not select bindings at this level, as all goals are grounded. Thus, it proceeds to select one of these goals as its focus and chooses the only relevant skill – for transporting a set of held balls. Because the skill's conditions are unsatisfied, ICARUS creates a subproblem to achieve them. Only one binding is possible, and this has only one unmatched condition – that the agent be holding as many balls as possible – so this becomes the new focus goal. The knowledge base does not include a skill that produces this as an effect, but it does have a definition for the concept. The system repeatedly chains backward off this rule, leading it to execute the primitive skill for picking up balls until the concept is satisfied. This in turn satisfies the original skill instance, causing ICARUS to move the robot to the target room. At this point, it repeatedly drops balls until all grippers are free, then shifts to moving balls to a different room.

As in Depots, we carried out a lesion study in the Gripper domain in which we disabled one or both of the heuristics for selecting skill instances. Experimental runs showed that, without their influence, the problem solver carried out considerable search, but their inclusion eliminated backtracking almost entirely. In this case, the two heuristics provided roughly the same search reduction when each was used in isolation, which may be related to the presence of high-level skills that produced distinctive effects.

5.3 Problem Solving on the Tower of Hanoi

Our third study dealt with the classic Tower of Hanoi puzzle (Newell & Simon, 1972). This task involves taking a set of disks on a source peg and moving them, one at a time, to a target peg. In this case we equipped ICARUS with seven concepts and a single primitive skill that let it move an unobstructed disk from its current peg to another peg that has no smaller disk. Because this task appears simple once the solution is revealed, many researchers view it as a 'toy' problem. But it can be quite challenging to both humans and computers, primarily because the conditions for moving a disk from one location to another complicate the solution considerably. Notably, our encoding does not assume the problem solver can observe directly that a disk is clear; the system must infer this situation from the position of the various disks on the pegs.

As with the previous domains, all goals in the top-level problem are grounded, but the system must still select one of these as its focus, which produces some search. However, the distinguishing feature of this task lies in negated conditions on the single skill that guard against moving a disk from or to a peg with a smaller disk on it. During problem solving, ICARUS repeatedly reaches a situation in which one of these negated conditions is unsatisfied. Lacking a skill that achieves it directly, the system chains off the negated concept, leading it to create a disjunctive subproblem based on negated versions of each subconcept. Because it must satisfy only one of the goal elements to solve a disjunctive problem, ICARUS selects one of them as its focus and, upon achieving it, returns to a higher-level problem. We will not give the details here, but, briefly, the system addresses a number of conjunctive and disjunctive subproblems before it reaches a state that solves the overall problem.

The two heuristics do not play a significant role on the Tower of Hanoi puzzle, but the ability to chain off the negated conditions of skills is crucial. The previous ICARUS problem solver module lacked this facility, which kept it from handling this task unless provided with a carefully crafted representation that avoided negations of defined concepts. We suspect that other problem-solving models would have the same difficulty in this domain, and we are not aware of any prior papers that report success on this task with the encoding that we have adopted.

5.4 Experience with Other Domains

We have also tested the new problem-solving module on a number of other classic domains from the literature. These include the blocks world, logistics planning, and tile-sliding puzzles. The blocks world involves placing blocks in a desired configuration, logistics concerns transporting objects to different locations, and tile sliding requires one to arrange a set of tiles through constrained movements. The ICARUS problem solver can handle problems from each of these domains, although – because the latter is nonserializable – the system requires higher-level skills to serve as macro-operators to find a solution. Combined with the results from the domains described earlier, these findings suggest that the new module has considerable breadth and generality, although in some cases it must carry out substantial search to find a solution, even with access to the new heuristics.

6. Related Research

Computational models of problem solving have a long history within AI and cognitive science, so it it not surprising that some ideas similar to those we have proposed have appeared before. We believe that we are the first to combine them in a unified framework, but we should still mention earlier efforts that have addressed closely related issues.

The PRODIGY (Carbonell et al., 1990) architecture, which also relies centrally on means-ends analysis, incorporates the notion of problems and subproblems in which goals are embedded. The Soar (Laird, Newell, & Rosenbloom, 1987) framework takes advantage of a similar idea[3] in its problem spaces, although the details differ and our approach comes closer to that in PRODIGY. However, the larger body of AI work on planning, which influenced the initial version of ICARUS, has generally ignored the organizing potential of problems to focus primarily on component goals.

Our distinction between problems and their formulations in terms of bindings appears to be more novel. PRODIGY also incorporated choice points for bindings, but this emphasized selecting among alternative instantiations for operators, rather than choosing among partial matches of problem statements against the current state. Other researchers (e.g., Riddle, 1991) have certainly examined the role of problem formulation in specifying problem spaces, but these treatments have dealt with representational issues rather than with decisions and processes that arise within the problem-solving activity itself.

Of course, heuristics for guiding problem-space search date back to the field's beginning, but these have typically incorporated domain-specific knowledge rather than generic calculations about relations between operators, goals, and states. Jones and Langley (2005) introduced an earlier ver-

3. Both probably borrowed the idea from structures used in Newell, Shaw, and Simon's (1960) General Problem Solver.

sion of our maximum goals heuristic to direct a means-ends problem solver, while Nejati (personal communication, 2010) has explored the minimal unsatisfied conditions metric in a similar setting, but we believe we are the first to combine them. To our knowledge, our use of disjunctive goals to handle chaining over complex negated conditions is entirely novel.

7. Directions for Future Research

Although ICARUS' new problem-solving module brings it closer to the capabilities observed in humans, it remains incomplete in some important ways that we should address in future work. One drawback lies in the systematic character of ICARUS' mechanism for problem-space search. Humans seldom search in a depth-first manner, at least partly due to memory limitations that make it difficult to retain an entire chain of problems. de Groot (1978) argues that human chess players instead use *progressive deepening*, a method that explores a single path at a time and retains information only on the first decision and the quality of the outcome. Future versions of ICARUS should incorporate similar methods for memory-limited, nonsystematic search during problem solving.

Another limitation is that not all human problem solving relies on backward chaining from goal elements. This approach dominates behavior on puzzles like the Tower of Hanoi and even novices' efforts to solve physics problems (Larkin et al., 1980). However, people sometimes carry out heuristic search in a forward direction, with little explicit influence from goals, especially in complex games like chess. Before ICARUS can offer a complete account of problem solving, it must have the ability to determine the direction of search dynamically. One response would build on the heuristics for selecting skills described earlier. Rather than giving the 'more goals' criterion precedence over the 'more conditions' heuristic, the architecture could compute their weighted average, with weights influenced by estimated branching factors in the backward and forward directions. Such an extended problem solver would still carry out means-ends analysis on some tasks, but on others it would exhibit forward-chaining behavior, even when the selected skills achieve none of the problem's top-level goals, and on still others it would pursue a mixed search strategy.

A third limitation concerns the role of insight in problem solving. The Gestalt psychologists studied behavior on many tasks that people find quite difficult until they manage to view them in a different light. Ohlsson (1992) has proposed an account of such insight in terms of restructuring or elaboration of the problem space that produces a simpler cognitive task. Fortunately, the new problem-solving module also suggests an approach to implementing this idea. Recall that, upon encountering a problem or subproblem, ICARUS selects a formulation of that problem based on partial matches of goals against beliefs and their associated bindings. In a simple sense, backtracking over this decision already supports a kind of reformulation. But suppose we adapted the architecture's conceptual inference process to operate over these structures to produce elaborated problem formulations and then used the result to select among them. Taken together, these mechanisms could provide an explanation for some insight effects, similar to the one MacLellan (2011) has proposed.

Because the final two proposals build directly on the extensions reported earlier in the paper, they provide further evidence, although mainly suggestive, that our elaborations on the theory of problem solving contribute to deeper understanding of this complex and important topic. We hope that future research along these lines and others will confirm and extend the explanatory power of the evolving ICARUS architecture.

8. Concluding Remarks

In the preceding pages, we reviewed the assumptions that underlie the standard account of human problem solving and their incarnation in ICARUS, a theory of the cognitive architecture. After this, we presented four extensions to this framework: the ability to represent and create problems as opposed to goals; the ability to encode and select among different problem formulations; the ability to chain over complex negated goals; and the ability to use domain-independent heuristics to guide problem-space search. These changes figure centrally in a new ICARUS problem-solving module.

We reported our experience with the new problem solver on three domains, in each case describing overall behavior and comparing it to that produced by its predecessor. In general, we found that the new system carried out far less search than the older one, and that in one case it solved problems that the other could not handle in principle. We noted that, although most of our extensions have appeared elsewhere in the literature, they have never been combined into a unified system, and that our framework also holds promise for explaining additional aspects of human problem solving.

Acknowledgements

This research was supported by Grant No. N00014-10-1-0487 from the Office of Naval Research. We thank Glenn Iba, Subbarao Kambhampati, Nan Li, Daniel Shapiro, and David Stracuzzi for useful discussions about the work.

References

Carbonell, J. G., Knoblock, C. A., & Minton, S. (1990). PRODIGY: An integrated architecture for planning and learning. In K. Van Lehn (Ed.), *Architectures for intelligence*. Hillsdale, NJ: Lawrence Erlbaum.

de Groot, A. D. (1978). *Thought and choice in chess* (2nd Ed.). The Hague: Mouton Publishers.

Jones, R. M., & Langley, P. (2005). A constrained architecture for learning and problem solving. *Computational Intelligence*, *21*, 480–502.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence*, *33*, 1–64.

Langley, P., Choi, D., & Rogers, S. (2009). Acquisition of hierarchical reactive skills in a unified cognitive architecture. *Cognitive Systems Research*, *10*, 316–332.

Langley, P., & Rogers, S. (2005). An extended theory of human problem solving. *Proceedings of the Twenty-seventh Annual Meeting of the Cognitive Science Society*. Stresa, Italy.

Larkin, J. H., McDermott, J., Simon, D. P., & Simon, H. A. (1980). Expert and novice performance in solving physics problems. *Science*, *208*, 1335–1342.

MacLellan, C. (2011). An elaboration account of insight. *Proceedings of the 2011 AAAI Fall Symposium on the Advances in Cognitive Systems*. Arlington, VA: AAAI Press.

Newell, A., Shaw, J. C., & Simon, H. A. (1958). Elements of a theory of human problem solving. *Psychological Review*, *65*, 151–166.

Newell, A., Shaw, J. C., & Simon, H. A. (1960). Report on a general problem-solving program for a computer. *Proceedings of the International Conference on Information Processing* (pp. 256–264). UNESCO House, Paris.

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

Newell, A., & Simon, H. A. (1976) Computer science as empirical enquiry: Symbols and search. *Communications of the ACM*, *19*, 113–126.

Ohlsson, S. (1992). Information processing explanations of insight and related phenomena. In M. Keane & K. Gilhooly (Eds.), *Advances in the psychology of thinking* (Vol. 1). London: Harvester-Wheatsheaf.

Riddle, P. (1991). *Automatic shifts of problem representation*. Doctoral dissertation, Department of Computer Science, Rutgers University, Piscataway, New Jersey.