Structural Transfer of Cognitive Skills

Dongkyu Choi (dongkyuc@stanford.edu)
Tolga Könik (konik@stanford.edu)
Negin Nejati (negin@stanford.edu)
Chunki Park (chunki.park@stanford.edu)
Pat Langley (langley@csli.stanford.edu)

Computational Learning Laboratory
Center for the Study of Language and Information
Stanford University, Stanford, CA 94305 USA

Abstract

This paper investigates a computational approach to transfer: the ability to use previously learned knowledge on related but distinct tasks. We study transfer in the context of an agent architecture, ICARUS, and we claim that many forms of transfer follow automatically from its use of structured concepts and skills. We show that ICARUS can acquire structured representations from domain experience, and subsequently transfer that knowledge into new tasks. We present results from multiple experiments in the Urban Combat Testbed, a simulated, real-time, three-dimensional environment with realistic dynamics.

Introduction

Many computational learning methods require far more training instances than humans to achieve reasonable performance in a domain. A key reason is that humans often reuse knowledge gained in early settings to aid learning in ones they encounter later. This phenomenon is known as *transfer* in cognitive psychology, where it has received far more attention than in AI and machine learning. Much of this research has focused on transfer of complex skills for tasks that involve action over time (e.g., Kieras & Bovair, 1986; Singley & Anderson, 1988). In this view, transfer primarily involves the reuse of cognitive structures, where the amount of shared structure has proved to be a good predictor for the degree of transfer in humans.

This paper reports on a computational approach to transfer that takes a similar perspective. We focus on the acquisition of cognitive skills from experience and on how transfer improves behavior on distinct but related tasks. We share with many psychologists the idea that transfer is mainly a structural phenomenon, rather than a consequence of statistical summaries or value functions. This suggests that transfer is linked closely to how an agent represents knowledge in memory, how its performance methods use these structures, and how its learning elements acquire this knowledge.

Theoretical commitments to representation, performance, and learning are often associated with the notion of a cognitive architecture (Newell, 1990). Thus, it seemed natural for us to study transfer in the context of ICARUS (Langley & Choi, 2006), an architecture that takes positions on each of these issues. We will maintain that ICARUS' commitment to relational, hierarchical, and composable knowledge structures, and to mechanisms for using and acquiring them, provide it with basic support for effective transfer. Moreover, we make the more radical claim that the architecture needs no additional mechanisms to exhibit many forms of transfer. We hold that most transfer requires no special processes beyond those already needed for other purposes.

We elaborate on these ideas in the sections that follow. First we present a virtual gaming environment that illustrates the benefits of reusing learned knowledge structures. After this, we review ICARUS' assumptions about representation, performance, and learning, along with the ways in which they support transfer. Next we evaluate our claims through results of specific studies with simulated physical agents. We conclude by reviewing related efforts on structural transfer and stating our priorities for future research on this topic.

An Example Domain

In this paper, we examine transfer between source and target problems within a single domain. We have chosen to phrase these problems in the Urban Combat Testbed[1] (UCT), a virtual 3-D environment that simulates an urban landscape, with real-time behavior and realistic dynamics. UCT contains one intelligent agent (controlled by ICARUS) and, at the moment, no adversaries. Our transfer tasks focus on navigation in the presence of physical and conceptual obstacles.

Figure 1 illustrates one such transfer task. The source problem calls on the agent to find a goal and surmount obstacles encountered en route (here, to duck under and climb over obstacles it has never seen). The target problem offers the agent the opportunity to reuse its knowledge about obstacles in a different order, assuming it is acquired and represented in a modular form. In addition, the agent can reuse learned knowledge about the map. The agent exhibits (positive) transfer if it improves its behavior in the target as a result of its exposure to the source, and zero or negative transfer if it does not.

We supply the ICARUS agent with minimal background to support transfer. On initialization, it has never encountered the specific objects or operators in the domain, and it has no prior knowledge of the map. However, it is initialized with useful concepts, such as a category for obstacles in general, a relation for blocked paths, plus categories for region centers and gateways (the UCT environment is divided into convex regions with passable

[1] http://gameairesearch.uta.edu/UrbanCombatTestbed.html

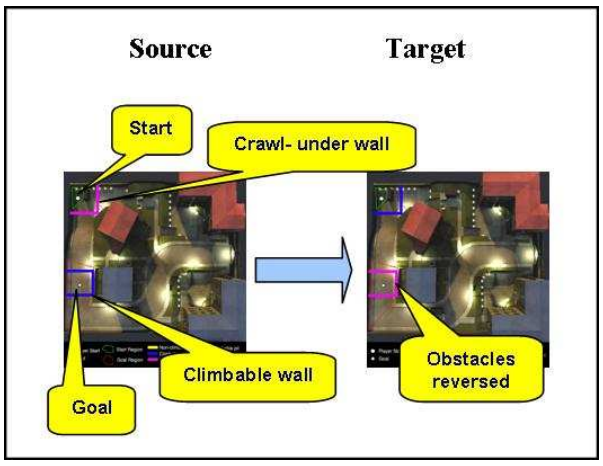

Figure 1: A transfer task in Urban Combat Testbed.

and non-passable boundaries). The agent understands the high-level goal (e.g., to find an item), and it possesses subgoals that organize search behavior. For example, it knows to overcome an obstacle in order to get a clear view of the destination, and to contain exploration within the region of the goal, once seen.

UCT is a challenging domain for both human and artificial agents. It is partially observable because the agent can only perceive nearby objects and regions, it involves uncertain action (e.g., the agent can attempt to jump over a wall but fall backwards into a ditch), and it is real time (imposing a strong constraint on agent decision making). This complexity demands a level of robustness in the mechanisms that produce transfer.

Transfer in ICARUS

ICARUS achieves transfer using a hierarchical and relational representation, which encodes knowledge in a general and composable way, a goal-driven and reactive execution mechanism, which allows flexible execution of the learned knowledge structures, and a relational learning mechanism, which acquires general knowledge from observed solutions as well as background knowledge. We will discuss each of these elements and describe in turn how they contribute to transfer.

Representation of Concepts and Skills

The ICARUS architecture makes several commitments in its representation of knowledge. First, it supports two different types of knowledge; concepts and skills. Concepts describe state, while skills are methods an agent can execute in the world under certain conditions. Both have a hierarchical structure, meaning that ICARUS can employ multiple layers of abstraction in describing the current state and the procedures for manipulating that state, respectively.

As shown in Table 1, concepts in ICARUS resemble traditional Horn clauses in first-order logic with negations. Primitive concepts like *in-region* provide state descriptions at the lowest level of abstraction using symbolic and numeric information directly extracted from

Table 1: Example ICARUS concepts.

```
(in-region ?self ?region)
  :percepts  (self ?self region ?region)

(climbable-gateway ?gateway ?object)
  :percepts  (gateway ?gateway) (object ?object)
  :relations ((totally-blocked-gateway ?gateway
                                       ?object)
              (feature-of-object ?object
                                 CLIMBABLE))
```

Table 2: Primitive and non-primitive ICARUS skills.

```
(clear ?gateway)
  :percepts ((gateway ?gateway
              dist1 ?dist1 angle1 ?angle1
              dist2 ?dist2 angle2 ?angle2))
  :start    ((close-enough-to-jump-type ?gateway))
  :actions  ((*jump-cmd (maximum ?dist1 ?dist2))
             (mid-direction ?angle1 ?angle2))

(crossable-region ?regionB)
  :percepts ((self ?self) (region ?regionB))
  :start    ((connected-region ?regionB ?gateway))
  :subgoals ((clear ?gateway))

(in-region-able ?me ?regionA ?regionB)
  :percepts ((self ?me)
             (region ?regionA)
             (region ?regionB))
  :start    ((in-region ?me ?regionA))
  :subgoals ((crossable-region ?regionB))

(in-region me region3004)
  :subgoals ((in-region-able me region3003
                             region3004)
             (in-region me region3004))
```

objects the agent perceives. Higher-level concepts, such as *stopped-in-region* and *climbable-gateway* have their basis in other concepts as well as primitive facts. The concept hierarchy provides relational, modular descriptions of the current state. It can also be used to represent a desired state, so concepts can express subgoals.

Taken together, ICARUS skills for a given domain are a specialized form of hierarchical task networks (Nau et al., 1999). A skill's head indexes it by the goals it achieves, and since goals are naturally represented by desired concept instances, skills are tied in to the concept hierarchy. Some achieve low-level concepts, while others address broad objectives. Table 2 shows some examples of skills in ICARUS. While primitive skills give simple methods using basic actions executable in the world, higher-level, non-primitive skills describe complex methods with multiple ordered subgoals. Since non-primitive skills specify subgoals, not the details of how these goals are achieved, an ICARUS agent can select the most relevant method for the given subgoal depending on the current situation.

ICARUS' relational and hierarchically composable representation of skills is crucial to its ability to transfer knowledge. In particular, the relational representation increases generality of the encoded skills, since they can apply in circumstances that are only qualitatively similar

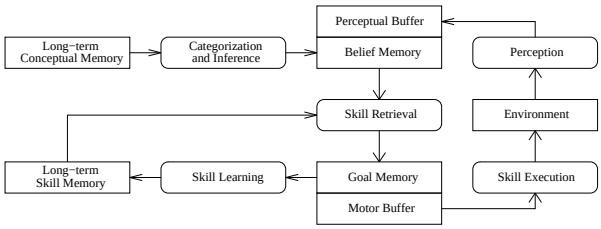

Figure 2: A schematic of memories and processes in the ICARUS architecture.

to the situations where the skills are acquired.

Moreover, a hierarchically composable representation lets skills apply in new circumstances, even if they are partially incorrect, inaccurate, or inapplicable. In these cases, the undesired subskills will be either relearned from new experience or dynamically replaced during execution with other subskills. For example, ICARUS can use a learned alternative that achieves the same goal, instead of an inapplicable subskill. We discuss how the architecture uses and learns hierarchically composable skills in the following sections.

Execution of Hierarchical Skills

The ICARUS architecture operates in cognitive cycles, spanning conceptual inference, skill selection, and physical execution (Figure 2). ICARUS derives its beliefs via a bottom-up matching process, initiated by objects that arrive in the agent's percepts. After it infers low-level concept instances based on these objects, inference for higher-level concepts follows to build a hierarchically organized belief structure for the time step. In contrast, ICARUS performs skill selection in a top-down manner, starting with the current goal. On each cycle, it finds a path from this goal through the skill hierarchy; each skill along this path is applicable given the current beliefs, with the terminal node being a primitive skill that ICARUS executes in the environment. This differs from traditional production systems, which require multiple cycles and use of working memory to traverse levels of an implicit goal hierarchy.

Since the architecture repeats this procedure on each cycle, ICARUS agents can react to their surroundings while pursuing goal-driven behaviors. Also, they can use new knowledge structures immediately, incorporating them for processing on the next execution cycle. Given a choice between two skills, ICARUS will prefer the one more recently learned, so agents can start with some knowledge but behave more intelligently as they acquire experience. The spatial search in UCT is a good example of this feature. Initially, the agent uses basic exploration strategy to search the environment. As it explores and discovers the geography of its world, it begins to employ the new map knowledge to guide further search towards the goal.

ICARUS' execution mechanism facilitates transfer by flexibly employing previously learned skills in three ways. First, it transfers learned skills to new problems by dynamically selecting and interleaving learned skills based on observed situations and achieved goals. Second, it combines skills learned from qualitatively different experiences when a novel situation has elements from these previous experiences. Finally, even if ICARUS does not have sufficient knowledge to directly solve a problem, it can transfer partially applicable skills from previous solutions and patch the knowledge gap by falling back on its default search skills.

ICARUS can achieve qualitatively different types of transfer by reusing high-level or low-level skills. For example, if a source and a target problem share abstract goals, it solves the target faster by transferring high-level skills. This occurs in UCT when the agent has learned how to clear a set of obstacles that block a goal. The agent uses the strategy acquired in the source (i.e., to approach the closest blocking object and overcome it) to tackle a different set of obstacles in the target, although the details depend upon the type of the obstacles and their relative configuration. On the other hand, ICARUS also transfers low-level skills by composing them in novel ways to solve the target problem. For example, if ICARUS learns to overcome a climbable wall in the source, it transfers that skill when it encounters a climbable wall in service of an unfamiliar route planning task, where the high-level goal might differ from that in the source problem.

One important requirement in transfer is to use previously acquired skills that are not completely correct or always applicable. This commonly occurs because conditions in the environment differ between source and target problems. For example, suppose the agent learns skills for navigating to the goal location in a UCT source problem. It partially executes those skills in the target, then abandons them in favor of exploration when it encounters a non-surmountable obstacle blocking the path. The agent reenters the skills when exploration brings it to some later step along its original path. This type of transfer results from ICARUS' reactive execution module, which uses only the relevant skills in the current environment.

Learning Hierarchical Skills

ICARUS acquires structured skills via an analytical learning mechanism that it invokes whenever the agent achieves a top-level goal. This mechanism inputs the goal plus a solution trace that achieves it, described as a sequence of observed states and selected actions. ICARUS generates an explanation of how the goal was achieved by interpreting this solution traces in the context of conceptual knowledge and action models. It does so by recursively examining goals, either decomposing them into subgoals using conceptual knowledge or explaining them in terms of the effects of primitive skills. The architecture converts the resulting explanation structure into hierarchical skills and add them to its skill memory. We have described this process elsewhere (Nejati et al., 2006) in more detail.

One distinctive property of this method is that it learns the structure of a skill hierarchy as well as the goals and conditions associated with each skill. This method is related to previous work on explanation-based

learning (Mitchell et al., 1986), but differs in that it does not compile away the explanation structure, but rather uses it to determine the structure of the skill hierarchy.

This learning mechanism facilitates transfer by associating a hierarchy of learned skills with the goals they achieve. As a result, the component skills can be used independent of the top-level goal that motivated their construction. For example, an ICARUS agent tasked to enter a building may find a solution where it jumps over a fence and then enters the building, viewed as sequence of primitive skills. The analytical learner creates a new skill to climb over a fence, as well as a higher-level skill that uses it along with primitives to reach the goal from the start location. The system associates the low-level skill (for fence climbing) with the goal for reaching a parameterized location, and it considers the component whenever a fence blocks a local goal.

The learning system also facilitates transfer by using relational background knowledge. It acquires skills that reference the agent's conceptual vocabulary, which provides flexibility in retrieving the skills. For example, an ICARUS agent in UCT may acquire a skill for reaching a goal it recognizes as collectively blocked (a built-in concept that matches when multiple objects impede a path). As long as the concept is true in a situation, the agent can apply the resulting skill regardless of the actual configuration of obstacles.

Evaluating ICARUS' Account of Transfer

The basic claim in this paper is that ICARUS' assumptions about representation, performance, and learning support transfer without need for any additional mechanisms. Moreover, both learning and transfer should occur at roughly the levels observed in humans, although we will not compare the architecture's behavior directly to the results of psychological studies here. We have already explained the ways in which ICARUS should produce such transfer, but this is different from demonstrating such effects.

To this end, we designed and ran a controlled experiment in the Urban Combat testbed. Our dependent variable was the time taken to solve a target problem that involved achieving a physical goal in the domain, and we studied the effects of two independent factors. One concerned whether the agent had experience solving five source problems (the transfer condition) or had no such experience (the nontransfer condition). The other involved the relationship between the source and target problems, which we discuss at more length below.

Source-Target Relationships

Analysis suggested a number of relationships between source and target problems that should support transfer of learned knowledge, six of which we focus on here. We consider two of these forms in detail and then briefly summarize the other four.

One source-target relationship, *abstracting*, involves sharing hierarchical solution structure, as the UCT scenarios in Figure 3 illustrates. Here the source and target problems involve the same start and goal locations, but

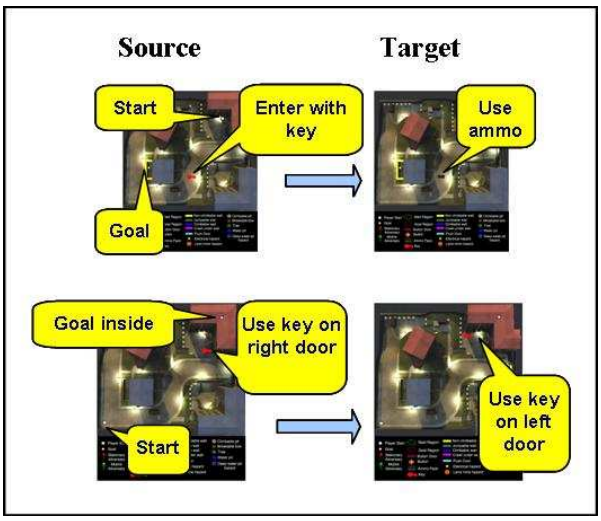

Figure 3: Two source-target pairs for abstracting.

the target requires solving a different subproblem to enter the chosen building. The problems can take advantage of the same route knowledge for navigating from start to goal, as well as the high-level solution structure composed of an initial route segment, an entry task, and a final approach. Other source-target pairs of this type require the agent to break into a building via different means or clear unfamiliar obstacles from its path. Problem pairs share the start and goal locations but exercise largely distinct sections of the UCT map.

We expect ICARUS to transfer the overall decomposition of a source problem's solution into the corresponding target. This capability also lets the agent reuse route information between non-corresponding sources and targets, as well as between target problems, to the extent it explores overlapping terrain while solving tasks. However, the system cannot share solutions for new subproblems introduced into target tasks, since they do not reoccur. Those component solutions (e.g., using ammunition to enter the building vs. a key) must be discovered in each target task, so they act to increase solution time and decrease performance.

Another type of relationship between source and target problems, *restructuring*, illustrated in Figure 1, requires the agent to use solutions to subproblems in different orders. Successful transfer lets the agent solve the target problem more rapidly because it has learned how to duck under a wall and climb over a wall in the source problem, independently of when those subtasks arise in the target. Other source-target pairs of this type – surrounding the start and end states with jumpable vs. unclimbable walls, boxes vs. pits, button operated vs. push doors, and water vs. electrical hazards – follow the same pattern.

We also examined four additional relationships between source and target problems:

- reuse partial solutions from a common start state in source and target problems, despite differing goals (*extrapolating*);

- repeatedly reuse solutions to subproblems from the source problem when working on the target problem (*extending*);
- dynamically compose solutions to problems from source problems to solve a more complex target problem (*composing*); and
- reuse the skills learned on the source task to solve a target problem, but apply different operators to novel objects that occur in the target (*generalizing*).

Our experiment examined ICARUS' ability to transfer learned skills from source to target problems that involved each of these six relationships. We felt that, if transfer occurred, it would provide evidence for the generality of the architecture's mechanisms.

Experimental Design and Results

Our experimental method involves presenting the ICARUS agent with a collection of source-target problem pairs. Each pair provides a known opportunity for transfer, while the set (called a scenario) supports some cross talk: a single source problem can enable transfer into multiple targets and the set of source problems supports transfer into any target.

As noted above, we ran the agent in both a transfer condition, in which it first solved a set of five source problems and then solved five target tasks, and a nontransfer condition, in which it solved only the five target problems. We ran the agent on six different sets of source-target pairs that reflected the relationships discussed above. We randomly varied the presentation order of the target problems to guard against effects of training order.

Figure 4 summarizes the results of the experiment by plotting the problem solution time for the nontransfer condition against the time for the transfer condition. Each x and y value represents the average score over 20 runs of the ICARUS agent, with different icons depicting distinct forms of source-target relationship. Entries above the diagonal line indicate that positive transfer occurred, while entries below the line reflect negative transfer. The figure shows that ICARUS generally exhibits positive transfer for most problems in each type of relationship. Moreover, this transfer occurs after experience with only five source problems, meaning that the rate of learning is roughly comparable to that observed in humans.

The key point is that ICARUS produces this transfer without any mechanisms above those required to draw inferences, execute skills that are indexed by goals, and acquire those skills from problem solutions. It does not require any additional processes to explain transfer across a variety of different source-target relationships. Our experimental study generally supports this claim about the emergent nature of transfer effects within the ICARUS architecture.

Related Research

Researchers in psychology have shown considerable interest in transfer, but their research has emphasized experimental studies. There have been a few computational models of this phenomenon, Kieras and Bovair (1986) and Singley and Anderson (1988) being two examples. Their models provided very accurate predictions for the degree of transfer in terms of the number of shared knowledge elements, but these were not coupled with learning mechanisms that could acquire the knowledge.

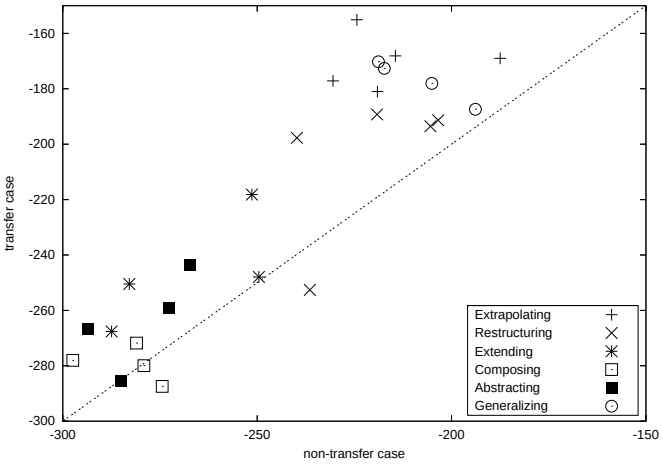

Figure 4: Solution times in seconds for transfer case plotted against those for non-transfer cases. Note that the axes are inverted so that higher scores indicate better performance.

In contrast, there have been several efforts on transfer in the machine learning literature. For example, Swarup and Ray (2006) discuss transfer in the context of neural networks, and Thrun (1996) considers the case in which an agent experiences many variations of a general task. However, these systems showed much slower learning than observed in humans, and made little contact with the psychological literature.

The approach we have described in this paper aims to model reuse of knowledge structure in the context of a cognitive architecture that incorporates psychologically plausible representations and mechanisms. There have been some previous results in the same spirit. For example, Langley (1985) investigated methods for learning search-control heuristics through discrimination of production rules, and Laird et al. (1986) demonstrated learning and transfer of macro-operators through chunking in Soar. More recently, Hinrichs and Forbus (2007) have discussed the transfer of planning and strategic knowledge among subproblems in a turn-based strategy game.

Directions for Future Work

Although our results to date have been encouraging, there clearly remains room for improving ICARUS' ability to transfer its learned knowledge. One avenue involves supporting more flexible execution of skills. Currently, the system executes skills in the environment as soon as they are applicable, but since skills acquired in one

setting can propose undesirable actions in another, they may lead to negative transfer. We are planning to add a module for lookahead search, constrained by the skill hierarchy, that would guard against this problem and thus improve transfer.

We are also investigating a method for making both inference and execution more flexible. This involves replacing ICARUS' current deductive inference module with one that relies on Markov logic (Richardson & Domingos, 2006), which combines first-order logical and probabilistic reasoning. This approach uses weights to soften the otherwise hard rules of a first-order knowledge base. Possible worlds that violate rules become more or less probable depending on evidence and the magnitude of weights. The result is an inference mechanism that is robust to error and uncertainty. This will let ICARUS transfer its skills more flexibly, in that it can select skills even when start conditions are likely but not deductively implied, as can happen in partially observable settings.

In this paper, we focused on forms of transfer that ICARUS can handle using its existing architectural mechanisms. But one implicit assumption of this approach is that the agent can use the same relational predicates to describe the source and target problems. This approach will not succeed in situations where the source and target problems have similar structure but have been encoded with different symbols. We aim to address this challenge by developing analytic methods that infer mappings between the representations used in the source and the target problem.

Concluding Remarks

Although structural transfer is an important phenomenon in human learning, there are few computational models that combine learning with transfer. In this paper, we described an agent architecture that can transfer skills learned in one setting to distinct but related tasks. We showed that the framework demonstrates this capability for a number of different relations between source and target problems, and we reported experimental results on a challenging testbed.

One of our key claims was that ICARUS can achieve transfer without requiring any mechanisms beyond those needed to represent, execute, and learn skills. Our experiments with UCT supported this claim and suggested that many types of transfer arise naturally from methods that can acquire relational, hierarchical structures. We analyzed ICARUS' ability to transfer and explained how its architectural commitments support this process. In the future, we hope to model additional forms of transfer that involve more complex relations between source and target problems.

Acknowledgments

The authors would like to thank Larry Holder and Michael Youngblood for the development and support for the Urban Combat Testbed, and John Laird and Nicholas Gorski for insightful discussions on cognitive architectures and agent development. Also, we would like to thank Dan Shapiro and Tom Fawcett for their contributions to scenario development and project evaluations, and Cynthia Matuszek and Blake Shepard for their work on domain refinement. This paper reports research sponsored by DARPA under agreement FA8750-05-2-0283. The U. S. Government may reproduce and distribute reprints for Governmental purposes notwithstanding any copyrights. The authors' views and conclusions should not be interpreted as representing official policies or endorsements, expressed or implied, of DARPA or the Government.

References

Hinrichs, T. R., & Forbus, K. D. (2007). Analogical learning in a turn-based strategy game. *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*. Hyderabad, India.

Kieras, D. E., & Bovair, S. (1986). The acquisition of procedures from text: A production-system analysis of transfer of training. *Journal of Memory and Language, 25*, 507–524.

Laird, J., Rosenbloom, P. & Newell, A. (1986). Chunking in SOAR: The anatomy of a general learning mechanism. *Machine Learning, 1*, 11–46.

Langley P. (1985). Learning to search: From weak methods to domain-specific heuristics. *Cognitive Science, 9*, 217–260.

Langley, P., & Choi, D. (2006). Learning recursive control programs from problem solving. *Journal of Machine Learning Research, 7*, 493–518.

Mitchell, T. M., Keller, R. M., & Kedar-Cabelli, S. T. (1986). Explanation-based generalization: A unifying view. *Machine Learning, 1*, 47–80.

Nau, D., Cao, Y., Lotem, A., & Muñoz-Avila, H. (1999). SHOP: Simple hierarchical ordered planner. *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence* (pp. 968–973). Stockholm: Morgan Kaufmann.

Nejati, N., Langley, P., & Könik, T. (2006). Learning hierarchical task networks by observation. *Proceedings of the Twenty-third International Conference on Machine Learning* (pp. 665–672). New York: ACM Press.

Newell, A. (1990). *Unified theories of cognition*. Cambridge, MA: Harvard University Press.

Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning, 62*, 107–136.

Singley, M. K., & Anderson, J. R. (1987). A keystroke analysis of learning and transfer in text editing. *Human-Computer Interaction, 3*, 223–274.

Swarup, S., & Ray, S. (2006). Cross-domain knowledge transfer using structured representations. *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. Boston, MA: AAAI Press.

Thrun, S. (1996). *Explanation-based neural network learning: A lifelong learning approach*. Boston, MA: Kluwer Academic Publishers.