Biologically Inspired Cognitive Architectures (2014) **8**, 93–99

Available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/bica

LETTER

A case study of knowledge integration across multiple memories in Soar

John E. Laird [*], Shiwali Mohan

Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2121, United States

Received 14 February 2014; received in revised form 18 March 2014; accepted 24 March 2014

KEYWORDS
Cognitive architectures;
Soar;
Situated interactive instruction;
Cognitive robotics

Abstract
Online perception, behavior, and learning in complex domains require an intelligent agent to quickly and reliably access different types of knowledge. A cognitive architecture, therefore, must implement a diverse set of memories that are optimized for storing, accessing, and learning these different types of knowledge. In this paper, we describe a complex Soar agent that uses and learns multiple types of knowledge while interacting with a human in a real-world domain. Our hypothesis is that a diverse set of memories is required for the different types of knowledge. We first present the agent's processing, highlighting the types of knowledge used for each phase. We then present Soar's memories and identify which memory is used for each type of knowledge. We also analyze which properties of each memory make it appropriate for the knowledge it encodes. These include procedural, semantic, and episodic knowledge, all of which play critical roles in the agent's ability to learn and extend its capabilities. We conclude with a summary of our analysis, and conclude that a diversity of memory systems and knowledge are useful for supporting general, integrated intelligence.

Introduction

Today's cognitive architectures provide one of the best examples of integration of multiple components to support the development of intelligent agents. The early version of Soar had only two architectural memory modules: working memory and production memory. Recently, we have significantly extended Soar (Laird, 2012), so that it includes multiple long-term declarative memories (semantic and episodic), a spatial short-term memory (SVS), as well as perceptual memories for robotic applications. These memories were added to expand ways in which knowledge can be stored and accessed. For example, episodic memory automatically records snapshots of working memory and supports

[*] Corresponding author. Tel.: +1 7346471761; fax: +1 7347631260.
E-mail addresses: laird@umich.edu (J.E. Laird), shiwali@umich.edu (S. Mohan).

http://dx.doi.org/10.1016/j.bica.2014.03.006
2212-683X/© 2014 Elsevier B.V. All rights reserved.

cue-based retrieval using partial match, whereas productions are learned based on problem solving activity and perform their actions whenever all their conditions match.

With this diversity, Soar agents can exploit the differences in memory systems to encode and access different types of knowledge. This is a case study of an agent that embodies unique and diverse types of knowledge that take advantage of the different characteristics of Soar's memory systems. The agent, called ROSIE[1], learns new concepts, words (nouns, adjectives, prepositions, and verbs), and problem definitions for simple games and puzzles such as Towers of Hanoi and Tic-Tac-Toe, from a human instructor in a shared real-world environment. ROSIE employs real-world visual sensing and motor control, limited language processing, dialogue management, problem formulation, internal problem solving and search, self-explanation, mental imagery, and multiple forms of learning. Except for perception, motor control, and some primitive parsing, all of these capabilities are realized through knowledge encoded in the architecture as opposed to through the addition of separate modules. Moreover, ROSIE's behavior and learning is all *in situ*, such that it is always ''on,'' with online real-time learning.

Many other cognitive architectures have similar sets of memories as Soar; however, none of them have published reports of running implementations of the complete suite of memories that Soar has or examples of agents that learn and use as diverse a set of knowledge types and memories as described here.

In this paper, we begin with a short description of our robotic domain, followed by an abstract overview of ROSIE's processing. The purpose of this overview is to identify the different types of knowledge that are learned and used by the agent. We then analyze the memory structures in Soar, identifying their distinctive characteristics in terms of how knowledge is encoded, stored, and accessed. Concurrently we describe and analyze which knowledge types are stored in each of the memory systems, identifying the features that each memory system provides and how they are appropriate for specific types of knowledge. Thus, we first present a *functional* analysis of our agent and then map those functions onto the memory *structures* in Soar. We conclude with a summary of our analysis.

The agent described in this paper has been previously described in Mohan, Mininger, Kirk, & Laird, 2012; Mohan, Kirk, and Laird (2013); however this paper provides a novel analysis of memory systems used in the agent, building on prior analyses of Soar memory systems (Derbinsky & Laird, 2010; Laird, 2012).

Domain

Our domain (as shown in Fig. 1) is a tabletop with an overhead Kinect camera that extracts low-level sensory data, and a robotic arm that can manipulate foam blocks of different colors, shapes, and sizes. The domain also includes four named locations: *pantry*, *stove*, *dishwasher*, and *garbage*. These locations have associated simulated functions. For example, the *stove* can be turned *on* or *off* and the *pantry* can be *opened* or *closed*. To act in the world, the agent sends discrete commands to the robot controller that executes a corresponding closed-loop policy. The commands include object manipulation: `pointTo(obj)`, `pickUp(obj)`, and `putDown(x,y)` and simulated location operations: `open(loc)`, `close(loc)`, `turnOn(stove)`, and `turnOff(stove)`. The instructor interacts with the agent through a chat interface. Messages are fed to the agent through the LG parser (Sleator & Temperley, 1991), which extracts syntactical structure. Messages from the agent are translated to natural language using pre-encoded templates.

Agent processing

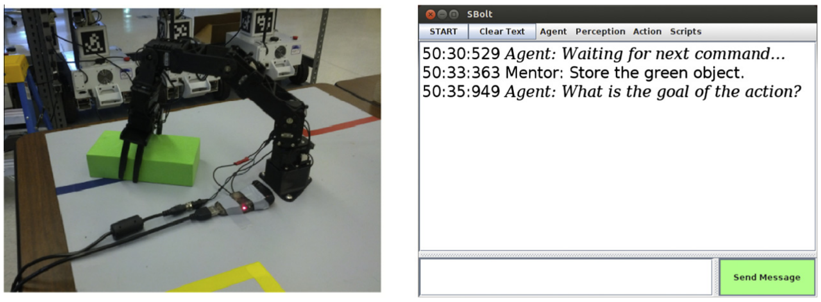
ROSIE is tasked with acquiring diverse kinds of knowledge — perceptual, spatial, semantic, action, task, and linguistic — through collaborative human—agent interactions. Our approach to learning — *situated, interactive instruction* — imposes requirements on how the knowledge is represented and accessed (described below) in ROSIE.

- *Situated:* The communication between the human expert and ROSIE is situated in the current state of the collaborative task. Through natural language-like utterances, the expert instructs the agent to perform novel tasks. ROSIE interprets the human's utterances within the real-world context by grounding words in visible objects, known spatial relationships, and known actions. Situated comprehension augments linguistic utterances with the knowledge of the shared perceptual state and shared knowledge of the domain and eliminates problems arising from under-specific and ambiguous utterances. To support such comprehension, the agent must use diverse kinds of knowledge and representations to extract the current perceptual, relational state from sub-symbolic sensory data; perform lexical analyses of utterances using its knowledge of English syntax; and generate situated interpretations of utterances by exploiting its knowledge of how lexical items are associated with the real-world state and its domain knowledge.

- *Interactive:* The communication is mixed-initiative and bi-directional, with both participants initiating interactions in accordance with their intentions and goals. The human and the learner form a system of joint learning that distributes the onus of knowledge acquisition between both partners. The human takes the initiative in identifying properties of the environment useful in learning novel concepts. The learner takes initiative in interpreting the instructions, analyzing its success and failures, and posing relevant questions. Knowledge representation and access should inform what questions the agent asks, when these questions are posed to the instructor, and how the replies are integrated in a comprehensive domain knowledge schema.

- *Distributed acquisition:* ROSIE is endowed with diverse capabilities including situated comprehension, learning, and complex, hierarchical task execution. To continually learn various concepts from instructions, ROSIE must induce knowledge that spans different modalities.

[1] RObotic Soar Instructional Entity.



50:30:529 *Agent: Waiting for next command...*
50:33:363 Mentor: Store the green object.
50:35:949 *Agent: What is the goal of the action?*

Fig. 1 The table-top workspace, robotic arm, and interaction chat window.

For example, the knowledge of spatial relationships is represented such that it is useful in not only recognizing useful relationships between objects in the visual sensory stream but can also be used for executing actions that establish the required relationship between target objects. This knowledge is also associated with prepositions such as *left-of* and therefore is useful in linguistic communication with the instructor. The ability to communicate about spatial relationships using prepositions allows ROSIE to learn complex action and task knowledge that incorporates the knowledge of those spatial relationships.

Fig. 2 shows ROSIE's overall processing structure that supports the above behavior. Various stages in processing are shown in rounded rectangles. Below we analyze the processing of each phase and its short-term (white rectangles) and long-term (gray rectangles) knowledge requirements.

Perceptual processing

Visual sensing of the environment occurs via a Kinect camera that provides color and depth data streams to the perception system. The perception system segments the scene into objects (blocks and locations). From examples provided through instruction, the agent learns to classify visible objects along three perceptual dimensions — color, size, and shape. A class in a classifier corresponds to a perceptual symbol. For example, a perceptual symbol R42 may correspond to the color red in the color feature space. Along with symbols that describe the perceptual features of an objects, the agent also learns to recognize and use spatial relationship predicates such as S22 that correspond to spatial prepositions such as *left-of*.

Through such perceptual processing, the agent generates an object-oriented, relational, perceptual state description of the environment from sensory data. This representation is useful for acting in the environment through object manipulation and for reasoning about goals of complex tasks.

Lexical processing

LG-Soar (Lonsdale, Tustison, Parker, & Embley, 2006) is a natural language component implemented as operators in Soar. It generates a syntactic parse of the utterance using a static dictionary and grammar. It uses part-of-speech tags and grammar of English to create a parse in the agent's working memory, identifying the useful content in the message. This parse is further categorized as verb-command, goal-description, descriptive-sentence, etc. based on its lexical structure. This categorization informs how the utterance is processed.

Interaction state management

After an utterance has been categorized, the agent uses knowledge of domain-general heuristics and the context of the ongoing dialog to associate intentions with the utterance. The intentions are useful in determining the next goal ROSIE should pursue. The goal can be to manipulate the environment through actions to establish certain state

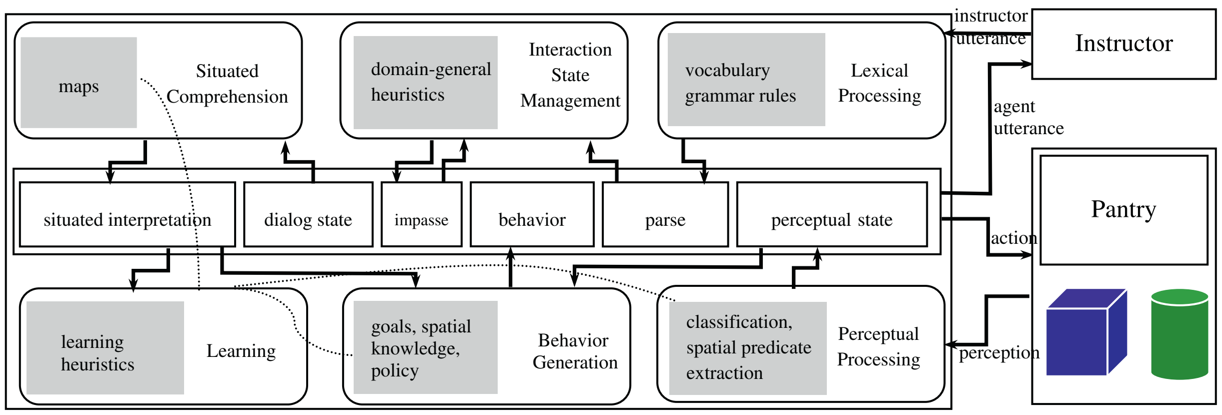


Fig. 2 ROSIE's processing structure. Gray boxes represent long-term knowledge accessed in each phase.

predicates or it can be to provide responses to the instructor's queries.

Situated comprehension

To gain useful information from instructions, ROSIE associates lexical items, such as referring expressions, prepositions and prepositional phrases, and verbs with their argument structures, to state descriptions and domain knowledge. Knowledge structures called *maps* encode how various lexical items are associated with perceptual descriptive and spatial symbols and action and task knowledge (*referents*). Maps are learned from interactive instructions. The comprehension process is formulated as memory search (or *indexing*) in which the agent attempts to retrieve maps for lexical items in an utterance. On successful retrievals, the agent composes the referents under the constraints derived from the retrieved maps, the current state of the environment, action models, and the context of the ongoing communication to generate a situated interpretation of the utterance. A failure in the search indicates that ROSIE does not possess knowledge to comprehend the utterance and triggers a learning phase in which the agent attempts to acquire the missing knowledge. If multiple interpretations are generated due to under-specific information in the utterance, the agent may initiate sub-dialogs to gather more information that is useful for generating unambiguous interpretations. This stage is described in detail in Mohan, Mininger, and Laird (2013).

Behavior generation

Apart from generating natural language-like responses to the instructor when queried about the environment, ROSIE also manipulates the environment in response to action commands. The agent is pre-encoded with primitive behaviors such as pick-up(object), put(object, location, relationship), and point-to(object). Through instruction, the agent learns complex behaviors that involve executing multiple primitive behaviors.

A behavior is represented by its *availability conditions* — a set of state descriptors that indicate that the behavior is available for execution; *application knowledge* — a set of rules that implement a policy; and *termination conditions* — a set of state descriptors, which when present, indicate that the goal of the behavior has been achieved. The policy is represented as rules that encode preferences for executing primitive behaviors given the current symbolic state description generated by the perceptual system and the goal the agent is pursuing. For example, for the behavior *store*(*object*) with a goal to place the object such that the spatial predicate *in*(*object*, *pantry*) is achieved, one policy rules creates a preference for *open*(*pantry*), if the state of the pantry is *closed*. If the pantry is *open*, a rule creates a preference to prefer the action *move*(*object*) *to the pantry*. Once the object has been successfully placed in the *pantry*, the action *close*(*pantry*) is preferred.

Learning

If failures occur in the situated comprehension or the behavior generation phases because the agent lacks the required long-term knowledge, an impasse arises. In response to an impasse ROSIE initiates interactions with the instructor to acquire the missing piece of knowledge. Multiple failures are handled incrementally until the agent resolves all impasses. From an impasse arising during situated comprehension and the ensuing interactions, the agent learns *maps* for nouns/adjectives (to perceptual attributes), spatial prepositions (to spatial relationships), and action verbs (to behaviors) leveraging the structure of interactions. Further semantic information of spatial compositions is also acquired from impasses arising during comprehension. Complex behaviors are acquired through interactions initiated in failures during the behavior generation phase.

To teach behaviors, the expert leads the agent through an example of executing the behavior by providing the sequences of primitive actions required to achieve the goal conditions of the complex behavior. Our learning approach uses chunking, which is a form of Explanation-Based Learning (EBL; Mitchell, Keller, & Kedar-Cabelli, 1986). A replay of the experience of executing the example guides the causal analysis of why the sequence of primitive behaviors was useful in achieving the goal of the complex behavior. The by-product of this causal analysis is the creation of selection rules that implement the behavior policy.

Summary

Across all of the processing, ROSIE uses a variety of distinctive forms of knowledge. Below is a summary of these, highlighting the types of mapping that are most dominant in each phase (each phase uses procedural knowledge to control its own processing; however, it is the behavior generation phase that procedural knowledge specific to the task being instructed is used).

- Perceptual processing
  Sensory data $\rightarrow$ perceptual symbols
- Situated comprehension
  Perceptual symbols $\leftrightarrow$ symbolic structures
  Symbolic structures $\leftrightarrow$ symbolic structures
- Behavior generation
  Symbolic state structure $\rightarrow$ actions
- Behavior learning
  Symbolic state structure $\rightarrow$ ensuing symbolic state

Soar memory system analysis

In this section we analyze Soar's memories from the perspective of the agent described above. The overall memory structure is shown in Fig. 3. See Laird (2012) for a complete description of Soar. As shown in the diagram, Soar has four different long-term memories and two different short-term memories. The diagram also shows the types of knowledge used in ROSIE. Colored blocks indicate innate (pre-programmed) types of knowledge, while the white blocks (with black lettering) indicate types of knowledge that are learned by the agent.

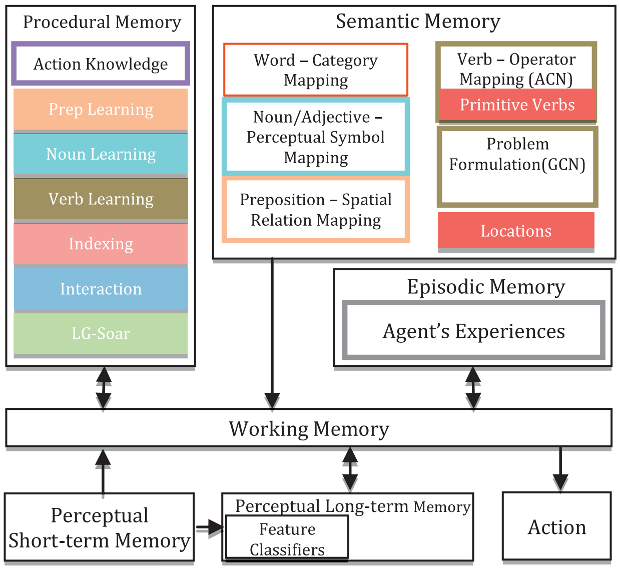


Fig. 3 The memory structure of the Soar and the knowledge types that are used in ROSIE.

Why does Soar need two different types of short-term memories and four different types of long-term memories? These memories do correspond to different memories found in humans, but what we are looking for is an independent justification from the function perspective: in what ways is each of the memories different so that they provide better functionality for encoding, storing, and retrieving different types of knowledge. In the previous section, we identified different types of knowledge and in this section we describe the characteristics of Soar's memories and how they satisfy the needs of those types of knowledge.

Before describing the specific memories, we provide some analysis on the key dimensions along which memories differ: encoding, storage, and retrieval. Our analysis is drawn in part from the distinctions made for episodic memory in Nuxoll and Laird (2010), but then focuses on the dimensions that differentiate the memories (whereas the original analysis focused on alternative designs for episodic memory).

Encoding is the act of representing the data in the memory. Memory systems differ in the representations they support, the content of what is encoded, and when and from where the information is encoded.

Storage is how a memory system maintains its data, including whether the contents of the memory change over time (such as through forgetting). The only dynamics for Soar's memories is that some have forgetting mechanisms, but these were not used in this agent.

Retrieval is the way other processes and memories access the information stored in the memory. Memory systems differ in whether data is always available or must be explicitly retrieved; whether retrieval is automatic or deliberate; if retrieval is explicit, what the cue is for retrieval; how a memory element is retrieved given a cue; and how the retrieved element is made available. For almost all of the memories, retrieval involves adding data to Soar's symbolic working memory.

We analyze memories in the order that they arise when the agent processes a command where it knows all of the words and can identify all of the referenced objects in the scene. For each memory, we give a general overview of its functionality, identify it functional characteristics, and then discuss which types of knowledge identified in the previous section are stored in it and why. We only analyze memories whose contents are open to change during the agent's processing, so that we do not analyze the fixed architecture processing (mostly implemented in C or C++) that operates over the memories, such as matching and firing rules or classifying perception using the contents of long-term perceptual memory.

Perceptual long-term memory holds the instances of training examples that are used to classify objects in terms of color, size, and shape and is a mapping from continuous sensory data to perceptual symbols. An instance is added whenever the agent sends a perceptual symbol to the memory. The memory is accessed for every object sensed in the environment, and if there is a successful match using its K-nearest-neighbor (KNN) algorithm, the appropriate perceptual symbol is associated with the object in perceptual short-term memory.

Although this has some similarities to matching rules in procedural long-term memory, this memory classifies continuous quantities as symbols based on the K nearest neighbors instead of via a symbolic match. Thus, it adjusts its category dynamically to new exemplars over time.

Perceptual short-term memory maintains a representation of the perceptual scene that includes both symbolic data (objects and their associated symbolic features described above) and continuous metric spatial information. These representations originate from perception or through a process called *projection* that allows the agent to imagine objects from symbolic working memory in the scene. An agent can selectively *extract* symbolic spatial relations (predicates) from the memory, such as the alignment of two objects in a plane. These relations are used for learning the meaning of prepositions. This memory, together with the associated processing for managing it, is called the Spatial Visual System (SVS; Wintermute, 2009).

SVS provides an interface between the continuous data in perception and the symbolic information in working memory, as well as a medium for mental imagery and spatial reasoning. Although this same information could be represented in working memory, the computational complexity of extracting predicates and projecting objects through rules would be significantly higher, so that in order to maintain reactivity, the agent would have to severely restrict the number of objects represented in a scene.

Working memory maintains symbolic relational representations of current and recent perceptual data, current goals, and the agent's interpretation of the current situation, which includes mappings between objects in the scene and internal symbols and words. It allows the agent to integrate data from perception with its long-term memories during language comprehension, dialogue management, problem solving, and acting in the world.

One weakness of using working memory to store large amounts of data is that the complexity of accessing procedural memory and the cost of storing memories in and retrieving memories from episodic memory increases as the amount of data in working memory increases. Thus, in order for an agent to maintain reactivity, it is not possible to store large bodies of knowledge in working memory (such as the knowledge of all the words it has learned). This weakness is one of the motivations for adding a separate long-term semantic memory.

Procedural memory contains a Soar agent's knowledge of how to select and perform both internal and external actions. This knowledge is encoded as rules. The conditions of the rules are continuously compared to working memory and when all of the conditions of a rule match, the actions make changes to working memory. Some changes are local to working memory, while others initiate retrievals from the long-term memories, interactions with SVS, or external actions in the agent's environment (such as moving the arm or communicating with the instructor). Thus, Soar's procedural memory maps from a situation to an action.

As shown in Fig. 3, ROSIE includes innate procedural knowledge for parsing sentences (LG-Soar), managing interaction with the instruction (interaction), and finding matches between mappings in semantic memory and the current instruction and the environment (indexing). Additional procedural knowledge guides the learning of words (nouns, verbs, and prepositions), storing mapping knowledge in semantic memory, and training the perceptual system based on categories extracted from instructions. Procedural memory also includes knowledge for acting in the world. Thus, it includes action knowledge for selecting and executing the actions associated with newly learned verbs.

The rules are a natural and efficient representation for this type of procedural knowledge. The efficiency is possible because the conditions for matching a rule are fixed and all conditions must match so that there is only one way the rule can match. These types of rules can be compiled into a discrimination network, so that the cost of matching scales well to very large numbers of rules (Forgy, 1982). Although it is possible to represent the same information in semantic memory, searching semantic memory so that every rule is matched every cycle is prohibitively expensive and does not scale as more verbs are learned.

Semantic memory stores context-independent declarative facts about the world, represented as symbolic graph structures. The agent can store working memory elements into semantic memory and it can retrieve them by creating a cue in a working memory buffer. The best match to the cue (biased by recency and frequency) is retrieved from semantic memory to working memory.

In ROSIE, semantic memory holds all of the mappings between words and their associated structures. There is a general mapping between a word and its category (noun/adjective, preposition, or verb), and then there are specialized mappings for each category. For nouns and adjectives, these are mappings from words to perceptual symbols. For prepositions, these are mappings between words and the combination of spatial relations that define the preposition. For verbs, there are mappings between the words and the structure of the Soar operator, which provides access to procedural knowledge.

ROSIE starts with some built in semantic knowledge, which includes mappings between words for locations and their spatial information, as well as the verb-operator mappings for the verbs that the agent starts with (pick up, put, and point).

Semantic memory makes it possible for the agent to access the information in each of these mappings in multiple ways. For example, when the agent perceives an object, it can be requested to describe it to the instructor using its acquired words (''This is a big red rectangle.''). In this case, the agent needs to map from a perceptual symbol to the appropriate adjective. Conversely, if the agent is given a command from the instructor that refers to the ''red'' object, the agent needs to determine which perceptual symbol the word ''red'' maps to. It is this ability to access a memory using multiple cues that distinguishes semantic memory from procedural memory. If this knowledge were encoded in procedural memory, it would be necessary to have a separate rule for every different way of accessing the memory. With semantic memory, when the memory is stored, the architecture does not need to know how it will be accessed. However, to avoid high computation costs, semantic memory is designed so that only one retrieval is attempted at a time; whereas procedural memory matches all of working memory and can fire multiple rules on every cycle.

Semantic memory has another feature that makes it very valuable in language processing, which is related to its biasing of retrievals to the most frequently and recently accessed memory elements. We have exploited this bias during sentence comprehension so that the instructor can use ambiguous references (such as ''that block'' or ''it''), and the Soar agent uses the retrieval bias in semantic memory to correctly identify the referenced object.

Episodic memory stores context-dependent records of an agent's experiences. It automatically takes snap-shots of working memory (episodes) and stores them in chronological order, enabling an agent to recall both the context and temporal relations of past experiences. An agent can deliberately retrieve an episode by creating a cue in a working memory buffer, and then retrieve subsequent episodes, replaying an earlier experience.

Other Soar agents have used episodic memory for many different purposes (Derbinsky, Li, & Laird, 2012). In ROSIE, episodic memory is used only to recreate the initial situation when it is learning a verb, and then replay its experience so that it can determine the causal structure of its actions. That causal structure is the basis for learning a generalized policy for executing a new verb. No other memory in Soar automatically maintains the data necessary for the experiential reply that episodic memory affords. The agent could deliberately store the same information in semantic memory, but that requires storing the complete state of working memory following every significant change to working memory, which would interfere with the agent's task performance.

Discussion

Our examination of ROSIE reveals a diverse set of knowledge and a correspondingly diverse set of memory mechanisms for storing those types of knowledge. The memories differ in their underlying representations, although given Soar's roots in symbolic processing, many of them employ symbolic representations. The inclusion of the short-term perceptual memory is in direct response to the need to reason about non-symbolic spatial representations for agents that interact with external environments.

Each of the symbolic long-term memories is optimized for different types of knowledge, with each trading off different capabilities to maintain reactivity in the face of scaling to large memories.

- Procedural memory is optimized for matching all rules against all of working memory, and then performing the actions of only those that match. Its tradeoff is that that it requires complete match of all conditions. Thus, each way of accessing a memory structure must be represented as a separate rule, whereas in semantic and episodic memory, the more permissive partial matching allows a single structure to be matched in many different ways. If arbitrary partial match were allowed for procedural memory, it would have little performance advantage over semantic memory retrievals, and would be distinguished only by the fact that it has separate actions from its conditions.
- Semantic memory is optimized for matching all subsets of a symbolic relational memory, and then providing complete declarative access to the complete memory. However, in contrast to procedural memory, it is matched against only a single cue, and not against all of working memory. Thus, it is not appropriate for fast execution of procedural knowledge. Nor is it appropriate for maintaining the context of specific experiences and the temporal structure of those experiences of episodic memory. However, it is well suited for representing general, context-independent mappings of the knowledge required in sentence comprehension, and scene interpretation.
- Episodic memory is optimized for storing and accessing the history of the agent's experiences. In contrast to semantic memory, it encodes specific instances of what existed in working memory as opposed to more general, context-independent knowledge. While episodic memory can be used to access prior memories from when general information was experienced, it requires knowing the context of how that general information was represented, making it difficult to create an appropriate cue.

To conclude, our research on interactive situated instruction has been a good case study for the integration of learning and using multiple forms of knowledge, and embedding of them in the memories of Soar. We lack a detailed evaluation or comparison of how these memories support these types of knowledge, although anecdotally, ROSIE demonstrates that they are sufficient for the types of instruction we have developed to date (Mohan et al., 2013). These include learning ten nouns/adjectives, six prepositions, and five verbs, in addition to ten different puzzles/games (Kirk and Laird, 2013). Derbinsky (2012) contains a rigorous evaluation of the performance of episodic memory and semantic memory across a wide range of tasks, but none have the variety of knowledge described here.

Acknowledgment

The authors acknowledge the funding support of the Office of Naval Research under Grant Number N00014-08-1-0099.

References

Derbinsky, N. (2012). Effective and efficient memory for generally intelligent agents. Ph.D. Thesis. University of Michigan.

Derbinsky, N., Laird, J. E. (2010). Extending soar with dissociated symbolic memories. *Proceedings of the 1st symposium on human memory for artificial agents* (pp. 31—37). Leicester, England: AISB.

Derbinsky, N., Li, J., Laird, J. E. (2012). A multi-domain evaluation of scaling in a general episodic memory. *Proceedings of the twenty-sixth AAAI conference on artificial intelligence*. Toronto, Canada.

Forgy, C. L. (1982). Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence, 19*, 17—37.

Kirk, J., Laird J. E. (2013). Learning task formulations through situated interactive instruction. *Proceedings of the second conference on advances in cognitive systems*.

Laird, J. E. (2012). *The soar cognitive architecture*. MIT Press.

Lonsdale, D., Tustison, C., Parker, C., & Embley, D. (2006). Formulating queries for assessing clinical trial eligibility. *Natural Language Processing and Information Systems*, 82—93.

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

Mohan, S., Mininger, A., & Laird, J. E. (2013). Towards an indexical model of situated language comprehension for real-world cognitive agents. *Proceedings of the Second Annual Conference on Advances in Cognitive Systems*.

Mohan, S., Kirk, J., & Laird, J. E. (2013). A computational model for situated task learning with interactive instruction. *International conference on cognitive modeling*.

Mohan, S., Mininger, A., Kirk, J., & Laird, J. E. (2012). Acquiring grounded representations of words with situated interactive instruction. *Advances in Cognitive Systems*.

Nuxoll, A. M., & Laird, J. E. (2010). Enhancing intelligent agents with episodic memory. *Cognitive Systems Research, 17—18*, 34—48.

Sleator, D., Temperley, D. (1991). Parsing English with a link grammar. Technical Report. Carnegie Mellon University.

Wintermute, S. (2009). Abstraction, imagery, and control in cognitive architecture. Ph.D. Dissertation. University of Michigan.