Available online at www.sciencedirect.com

SciVerse ScienceDirect

Cognitive Systems Research 17–18 (2012) 34–48

Cognitive Systems RESEARCH

www.elsevier.com/locate/cogsys

Enhancing intelligent agents with episodic memory

Action editor: Vasant Honavar

Andrew M. Nuxoll *, John E. Laird

University of Michigan, 2260 Hayward Street, Ann Arbor, MI 48109-2121, USA

Available online 31 October 2011

Abstract

For a human, episodic memory is a memory of past experiences that one gains over a lifetime. While episodic memory appears critical to human function, researchers have done little to explore the potential benefits for an artificially intelligent agent. In this research, we have added a task-independent, episodic memory to a cognitive architecture. To frame the research, we propose that episodic memory supports a set of cognitive capabilities that improve an agent's ability to sense its environment, reason, and learn. We demonstrate that episodic memory enables agents created with our architecture to employ these cognitive capabilities.
© 2011 Elsevier B.V. All rights reserved.

Keywords: Episodic memory; Cognitive architecture; Soar

1. Introduction

One advantage that humans have over current Artificial Intelligence (AI) systems is a personal history of specific events that they can draw upon to improve their decision making and learning. This episodic memory was first described in depth by Tulving (1983, 2002). Tulving's focus was phenomenological and in particular distinguished episodic memory from semantic memory. Episodic memory provides humans with the ability to remember where they have been, what they have sensed, and what actions they have taken in various situations. This knowledge of the past is invaluable for acting in the present. Episodic memory supports a wide range of cognitive capabilities from keeping track of the world outside immediate perception, to allowing retrospective learning on previously encountered situations. Certainly, there is evidence that human cognition is severely crippled by the loss of episodic memory and the difficulties that amnesiacs face have been documented (Tulving, 2002) and were dramatically portrayed in the movie Memento (Nolan, 2000).

As in any learning system embedded in a performance system, episodic memory involves: the capturing and encoding of experience in an internal format; storing that experience in a knowledge base for future use; retrieving knowledge in the future when given an appropriate cue. In addition to supporting these fundamental operations, there are additional functional requirements that Tulving (1983) identified that distinguish episodic memory from other memory and learning mechanisms:

- *Automatic*: The system creates new memories automatically without the agent deciding to do so. The underlying assumptions are that: (a) deliberately deciding which situations to remember can interfere with task-based reasoning and (b) it is unlikely that the agent can effectively determine which experiences will be relevant to future decisions.
- *Autonoetic*: A retrieved memory is distinguished from current sensing, so that an agent does not confuse a retrieved memory with the current situation.
- *Temporally indexed*: Because an episodic memory describes a particular, unique moment in time, some temporal information is a part of any episodic memory and

* Corresponding author. Present address: University of Portland, MSC 145, 5000 N Willamette Blvd., Portland, OR 97203, USA. Tel.: +1 503 943 7688; fax: +1 503 943 7316.

E-mail addresses: nuxoll@up.edu (A.M. Nuxoll), laird@umich.edu (J.E. Laird).

1389-0417/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.cogsys.2011.10.002

can also be part of an episodic memory cue. This need not be an exact time but it should convey a sense of the relative time of the episode with respect to other episodes.

This paper presents our progress toward creating a general purpose episodic memory within a cognitive architecture that supports the creation of general AI agents, that is agents that use large bodies of knowledge, continually learn from experiences in their environment, pursue multiple diverse tasks, and exist for extended periods of time. Although there has been sporadic research on episodic memory within AI in the past, there has not been research on task-independent episodic memories that support a wide variety of cognitive capabilities within a cognitive architecture.

Thus, our research involves determining the requirements for an episodic memory; designing, implementing, and integrating an episodic memory system within a cognitive architecture; and exploring the capabilities supported by such an integration. The emphasis of our research has been to create a computational system with the most important features of episodic memory so that we can develop and evaluate not just an episodic memory module, but an integration of that module within a cognitive architecture in which we can build agents. This paper describes our progress to date on this work. While we have made considerable progress, our episodic memory architecture is far from complete. For example, it does not include memory consolidation, forgetting, interference, priming, generalization across episodes or specific models of time. Our architecture does support effective encoding, storage and retrieval and we have used it to create agents for a variety of tasks. Our research suggests that episodic memory enhances the performance of AI agents and may be a "missing link" in current cognitive architectures, enabling a gamut of cognitive capabilities.

2. Cognitive capabilities

The focus of our research is to investigate whether episodic memory can support high-level cognitive capabilities beyond the simple recall of previous events. We define a cognitive capability as a beneficial ability or pattern of behavior that can be indirectly observed via the actions of intelligent agents. We do not claim that episodic memory is the sole means for achieving these capabilities but we claim that episodic memory can play an integral role is realizing these capabilities. Our research has focused on demonstrating multiple capabilities with a single architecture in a limited set of tasks set within two distinct environments. Our hypothesis is that episodic memory is integral to supporting the following cognitive capabilities, and in Section 7, we support this claim with example implementations.

- *Action modeling*: If an agent can recall the outcomes of its past actions, it can use that information to predict the outcomes of those same actions in the present. By recalling situations similar to the present from its past, an agent can use episodic memory to predict the immediate changes in its environment that will result from a given action.
- *Decision-making based on past experiences*: A history of prior successes and failures can be used during planning to guide decision making. Episodic memory can provide access to memories of similar situations that can help direct the agent to its goal and away from failures.
- *Retroactive learning*: Often, it is not possible to learn while an event is occurring because the agent lacks the specific information or resources that it needs to learn. For example, an agent in a real-time environment may not have time to apply an iterative learning algorithm while it is performing a task. Episodic memory allows previous experiences to be relived or rehearsed once the resources are available.
- "*Boost*" *other learning mechanisms*: An episodic memory store provides a wealth of data for training other learning mechanisms potentially allowing the agent to speed the rate at which it learns in new situations and even learn how to behave in situations it has never encountered before.
- *Virtual sensing*: In general, an agent's sensing is limited to the current situation: what is available directly from perception. Memory of recent events or situations can greatly expand an agent's ability to interact with the world outside. Episodic memory expands an agent's sensing by providing memories of areas beyond its immediate perceptions.

There are potentially many more cognitive capabilities that episodic memory enables, such as prospective memory (Kliegel, McDaniel, & Einstein, 2007) where an agent decides to perform future activity, such as stopping at the store on the way home from work, and then uses episodic memory to recall that plan at an appropriate time in the future. One of our goals for future research is to investigate more of the capabilities that may be possible through the addition of episodic memory to a cognitive architecture.

3. Related work

Within Artificial Intelligence, episodic memory research is closely related to case-based reasoning (CBR) (Kolodner, 1993; Schank, 1999). In a typical CBR system, each case describes a problem that the agent faced and a specific solution to that problem. When a new situation arises, the agent retrieves a stored case and adapts its solution to the new situation. Cases are usually described by a fixed number of task dependent fields, which are designed by a human, making them task dependent. Nonetheless, research in CBR highlights some important research that is equally relevant to episodic memory. In particular, Goodman (1993) describes using a CBR system's case library to predict future events. This prediction is analo-

Table 1
Summary of episodic memory research.

	Action modeling	Decision making based on past experiences	Retroactive learning	"Boost" other learning mechanisms	Virtual sensing
Goodman (1993)		x			
Vere and Bickmore (1990)					x
Ho, Dautenhahn, and Nehaniv (2003)					x
Tecuci and Porter (2007)		x			
Ram and Santamaría (1997)	x				
Nuxoll and Laird (this research)	x	x	x	x	x

gous the cognitive capability we call "learning from past successes and failures."

Vere and Bickmore (1990) implemented a limited episodic memory of events for their Basic Agent. Because of its peripheral nature to the goals of their research, the effectiveness, efficiency and completeness of their agent's episodic memory implementation was limited, but it demonstrated the potential of episodic memory for integrated agents. Vere and Bickmore's agent was particularly focused on the cognitive capability we call virtual sensing.

Ho, Dautenhahn, and Nehaniv (2003) developed an agent that uses a task-specific episodic memory to backtrack to previously seen locations. Their work demonstrates the cognitive capability we describe as virtual sensing; however, the episodic memory is specially designed for the specific tasks of remembering the location of entities that the agent has encountered.

Most recently, Tecuci and Porter (2007) developed a generic episodic memory module that they successfully used for multiple tasks including planning and physics problem solving. This system bears some similarities to the one we present here, but it is not embedded in a general architecture. Tecuci and Porter's work focused on using episodic memory to do planning by learning from past successes and failures.

In the field of machine learning, instance-based techniques provide a flexible way to learn the best action for the agent (Atkeson, Moore, & Schaal, 1997; Ram & Santamaría, 1997; Sheppard & Salzberg, 1997). The instance database used by these techniques is a simplified, task-specific form of an episodic memory. Nonetheless, it provides evidence that giving an agent a personal history can allow it to demonstrate the cognitive capability we call action modeling. Table 1 summarizes each of these bodies of research and the cognitive capabilities they demonstrate. The goal of our research is to provide a task independent implementation that supports all of these capabilities, as shown in the final line in the table.

4. The Soar architecture

Our research does not consider episodic memory in isolation, but instead we explore episodic memory as a component of a broader cognitive architecture. Specifically, our episodic memory module is integrated within the Soar cognitive architecture (Laird, 2008). Soar is a general cognitive architecture that has been used to model a wide variety of phenomena (Newell, 1990) as well as to create large knowledge-intensive AI agents (Jones, Tambe, Laird, & Rosenbloom, 1993). Soar shares many features in common with other architectures such as ACT-R (Anderson, 2007), Icarus (Langley, 2006), Clarion (Sun, 2006), and EPIC (Kieras & Meyer, 1997), so that although our research on episodic memory uses Soar, our overall design is relevant to creating episodic memories in those other architectures.

Fig. 1 depicts a high level view of the Soar architecture prior to the integration of episodic memory. As described in Laird (2008), Soar has recently been expanded to include episodic memory as well as semantic memory, reinforcement learning, and mental imagery; however, the work described here focuses on the integration of episodic memory with the components shown in Fig. 1.

Soar has a short-term working memory and long-term procedural memory. These are depicted in the large rounded rectangles on the right. Working memory is a short-term declarative memory that encapsulates the agent's current state including external sensing, the results of internal inferences, selected actions, and active goals. The working memory consists of identifier–attribute–value triplets called working memory elements (WMEs). The value of one WME can be the identifier of another WME so that these elements are linked together in a graph struc-

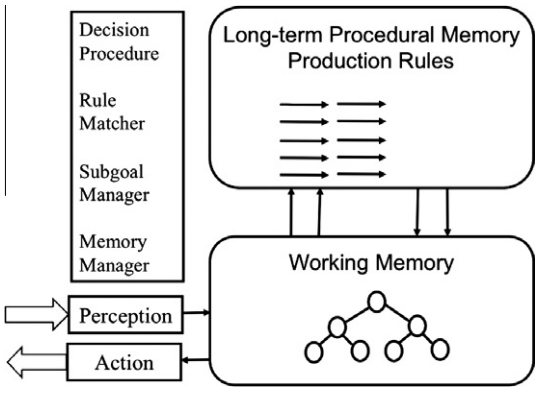

Fig. 1. The soar cognitive architecture.

ture. Soar's procedural memory consists of if–then production rules. If the conditions of a production are satisfied by the contents of working memory then that production fires and performs its actions which consist of creating or removing elements from working memory. All rules that match fire in parallel.

Soar has a basic control loop where it proposes, evaluates, selects, and applies actions (in other descriptions of Soar, these actions are called operators). Proposal rules create representations in working memory of the actions that are available in the current situation. Evaluation rules test for proposed actions and create preferences for selecting between them. A fixed decision procedure interprets the preferences and selects a single action. Once an action is selected, application rules apply the selected action, either by making changes to working memory, or initiating external actions. Only one action can be selected and applied in a given cycle, making actions the locus of decision-making.

If the knowledge encoded in rules is insufficient to select or apply the next action, Soar encounters an impasse. In response to an impasse, the architecture creates a substate to resolve the impasse. The same process of selecting and applying actions is used to resolve the impasse by making changes to the original, parent state that allow processing to continue in that state. Once an impasse is resolved in a particular state, all substates are removed and processing continues in the original goal. An impasse is also an indication that the Soar agent needs to learn. If the agent does not know what to do next, but subsequently discovers an appropriate action in a subgoal, then it has learned something. Soar employs a learning mechanism called chunking that converts the learned knowledge into a new production that will fire in similar situations in the future, eliminating the need for an impasse.

The version of Soar used by this research includes working memory activation[1] (Nuxoll, Laird, & James, 2004). The purpose of activation is to indicate the importance of a WME to the current processing, and so the activation level is based on the recency and frequency of creation and access of the WME element as follows:

- A WME receives an initial, default activation level when it is created.
- Any time that a WME is tested by a production that fires, its activation level is increased.
- Any time the agent attempts to add an already existing WME, the existing WME receives an increase in activation.
- WME activation decays exponentially over time. The rate of decay is determined by the frequency and recency with which it has received activation increases.

[1] The work presented in this paper is an extension to Soar 8. Soar 9 now contains a newer implementation that is optimized for faster execution and better scaling with large memory but lacks the activation-based biased retrieval described here (Derbinsky & Laird, 2009).

5. Episodic memory implementation

Fig. 2 depicts the integration of our implementation of episodic memory with the Soar architecture. (Compare this diagram to Fig. 1.) The central rounded-rectangle in the figure contains Soar's working memory.

Our implementation adds an episodic learning module that monitors the agent's behavior. At prescribed times, the module records a new episodic memory. Each new memory is effectively a snapshot of working memory taken at the time of recording. This collection of memories forms a new episodic memory store (depicted in the upper right of Fig. 2) that can be queried by the agent.

To illustrate the episodic memory system, we use a simple simulated environment, called Eaters, that is similar to Pac-Man. In Eaters, a Soar agent controls an "Eater", which moves around a small grid-based map consuming food. (More details are provided in Section 6.1) Fig. 3 shows an Eater and the portion of a map it can directly sense.

As shown in Fig. 4, the agent's sensing is represented in its working memory as a symbolic graph structure. At the center of the figure is a node that is the representation of the agent's current position, which includes features about the shape of the agent and its color. This node has additional relations (north, east, south, west) with other nodes that represent the cells in the four cardinal directions from the agent. These nodes have information about their contents as well as relations to all of the nodes they are adjacent to. The icons in each of the nodes are not represented in working memory and are included to make it easier to draw a correspondence between the nodes and Fig. 3.

An agent's information about its current situation is called its "state", which is maintained in working memory. The state of an agent contains its sensory information as well as additional information (not shown in Fig. 4) about the agent's potential actions, its goals, and the results of any inferences it has made about the situation. The agent's state that is the basis for creating an episodic memory, and the processing associated with episodic memory is decomposed into the following major phases:

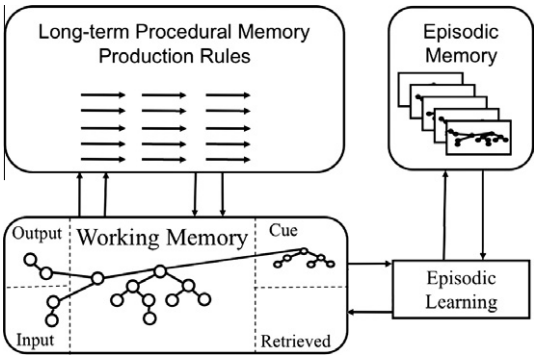

Fig. 2. Episodic memory implementation architecture.

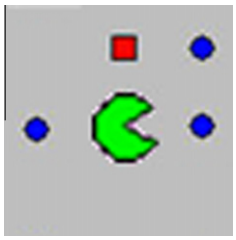

Fig. 3. Perceptual field of an Eaters agent.

- *Encoding*: how the state is captured and stored as an episodic memory
- *Storage*: how an episodic memory is maintained
- *Retrieval*: how a memory is retrieved.

The following sections describe the steps involved in each phase and the design decisions we made for that phase.

5.1. Encoding

Encoding involves capturing a memory to be stored in episodic memory. For many learning mechanisms, this stage can involve significant processing to determine the contents of the memory, possibly performing generalization over other long-term memories. However, in our approach, encoding for episodic memory is a fast, simple learning mechanism that does not interfere with the ongoing reasoning and problem solving of the agent. There are three basic design decisions that must be addressed in determining how and when encoding is performed.

5.1.1. Encoding initiation: when is a new episode encoded?

The agent could determine this deliberately, deciding when to record an episode based on features of the task or environment, but one of the functional requirements for episodic memory is that encoding is automatic. In our implementation, a new memory is encoded each time the agent takes an action in the world. In our experiments, alternative implementations, such as recording every processing cycle, had little impact on performance except to change the total number of episodes.

5.1.2. Episode determination: what are the contents of an episode?

The contents could include perception, motor commands, and information from internal processing. In our implementation, an episode consists of the agent's state, which includes its input (sensing), internal data structures and output (actions in the world), but currently excludes episodic memory cues and retrievals. Including these makes it possible for individual episodes to grow quite large as subsequent episodes could contain previous memories which could, in turn, contain a previous episode, and so on.

Working memory elements with activation below a given threshold are excluded to eliminate elements that were most likely irrelevant to the current processing.

In our Eaters example, if the agent decides to move west from its current position, an episode is recorded that con-

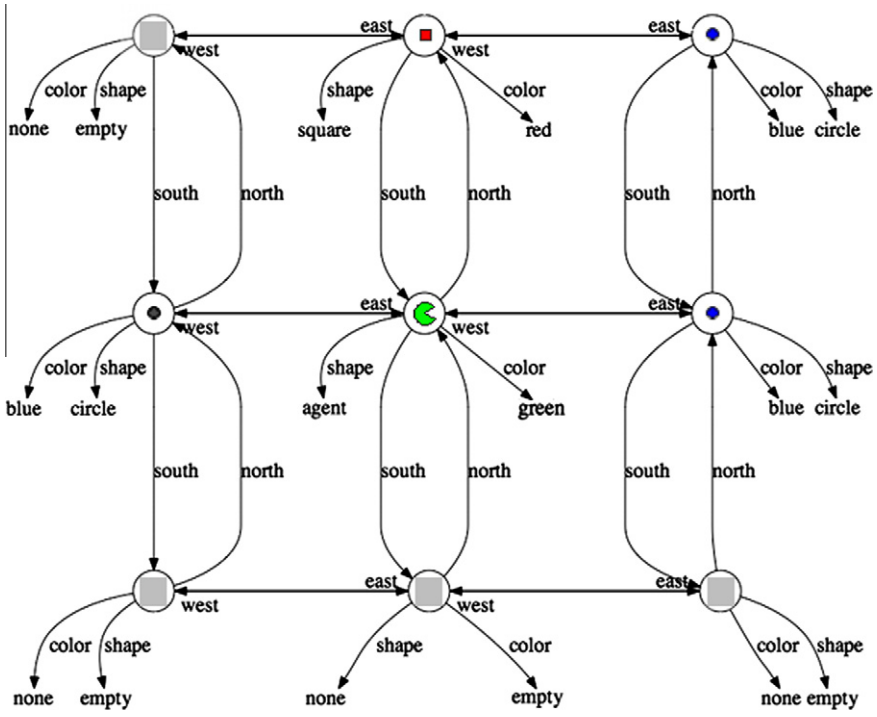

Fig. 4. A portion of an agent state as represented in working memory.

tains all of the information depicted in Fig. 3 as well as other information, including the fact that the agent has issued a "move west" command.

5.2. Storage

Once an episode has been identified, it must be stored. Two issues must be addressed in terms of storage.

5.2.1. Episode structure: what is the structure of the episodic memory store?

Conceptually, episodic memory consists of a sequence of individual episodes; however, the underlying implementation is optimized to support efficient and stable storage as well as efficient episode retrieval. Our approach takes advantage of the fact that few items in working memory change from episode to episode so it stores only the changes between successive episodes (Derbinsky & Laird, 2009).

5.2.2. Episode dynamics: do the episodes in the store change over time?

Once an episode is stored, it is possible that it could change over time. For example, episodes could be forgotten to minimize storage and some process could analyze existing episodes, attempting to detect commonalities and generalizations. In our implementation, the episodes do not change over time, though the benefits of this have been discussed in (Nuxoll, Tecuci, Ho, & Wang, 2010).

5.3. Retrieval

An agent uses episodic memory by retrieving an episode from long-term memory and then using the contents to influence decision making. Answers to the three questions below define the structure of our approach to retrieval.

5.3.1. Retrieval initiation: when is an episode retrieved?

Is retrieval initiated deliberately by the agent or can the retrieval occur spontaneously? In our implementation, retrieval is deliberate and is triggered when the agent constructs a cue in working memory. Spontaneous retrieval eliminates the ability of the agent to select a subset of the state for a cue, which is critical for focusing the retrieval on specific aspects of the current state. In addition, spontaneous retrieval puts a significant computational load on the system to continuously find an episode that matches the current state.

5.3.2. Cue determination: how is the memory cue specified?

The cue is a partial description of an episode that is matched against the stored episodes. In our implementation, the agent constructs a cue in a reserved area of working memory. For our implementation, we defined two reserved areas in working memory, which are delineated with dashed lines in Fig. 2. These are labeled "Cue" and "Retrieved" in the figure.

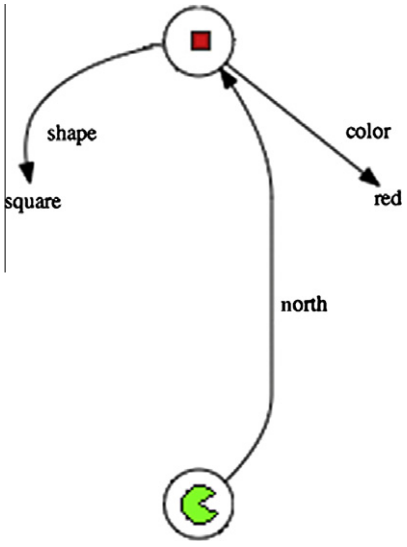

Fig. 5. An example episodic memory cue.

When an agent attempts to retrieve an episode, it creates a memory cue in the "Cue" area. This cue is typically a relatively small set of features that the agent would like to find in a previous episode. Fig. 5 depicts an example memory cue for an agent that is attempting to remember the last time it saw a red square to the north.

Our implementation also allows the agent to restrict the retrieval based upon its temporal index. For example, the agent can retrieve an episode that occurred sometime before a given time or an agent can retrieve the episode that occurred immediately following a given episode.

5.3.3. Selection – given a cue, which episode is retrieved?

Conceptually, the cue is compared to all of the stored episodes, and the one with the best match score is retrieved, unless the match score is below a threshold, in which case no episode is retrieved and a failure reported. The actual implementation takes advantage of many optimizations to avoid an exhaustive comparison of cue to episodes (Derbinsky & Laird, 2009). We have evaluated a variety of metrics for determining a match score. The most successful implementation weights two factors: (a) the cardinality of the match, which is the number elements that the cue and memory have in common and (b) the activation level of the working memory elements in the cue that match the memory (Nuxoll & Laird, 2004).

5.3.4. Retrieval: how is a retrieved episode represented to the agent?

In our implementation, the episode with the best match score is reconstructed in the reserved area of working memory that is labeled "Retrieved" in Fig. 2 in the same format in which it was in when first encoded. In addition, we add meta-data about the retrieval: a unique temporal index indicating when the memory was encoded and information about the strength of the match between the retrieved

memory and cue. This allows the agent to determine if the match was perfect or partial.

As this episodic memory system was built, we used it to implement a set of cognitive capabilities to explore the value of a task-independent episodic memory system in a cognitive architecture. The implementation evolved in parallel with the architecture as different design variations were explored. The remainder of this paper focuses on the cognitive capabilities and how they were implemented using episodic memory.

6. Test environments

To explore the efficacy of this system and compare alterative implementations, we created agents, encoded as knowledge in Soar, to perform specific tasks set in two different environments. Given that task independence is a goal for our implementation, it was essential to have multiple tasks and, ideally, multiple environments. This section describes the two environments we used for this research.

6.1. Eaters

The first domain is Eaters and a more complete depiction of it is shown in Fig. 6. An Eater is an agent in a 16×16 grid world. Each cell in the grid is either empty or contains one of three items: a wall, normal food worth 5 points, or bonus food worth 10 points. The Eater is able to move in each of the four cardinal directions unless there is a wall in its way. Each time it moves into a cell containing food, it automatically eats the food and receives the associated points. When an Eater leaves a cell, the cell becomes empty. The Eater's goal is to get the highest score it can, as fast as it can. The Eater's sensory input includes the contents of nearby cells and its current score. Fig. 6 contains an image of the Eaters playing board on the left as well as a graphical depiction of the input available to an Eater on the right.

Although this domain is simple, the perceptual features are repeated over and over again in varying patterns. This is, in effect, a hostile environment for an episodic memory agent because all the agent's episodic memories are similar but with small, significant differences. In addition, the simplicity of the environment and relatively fast pace of the task requires that many episodic memories be recorded, making it easy to quickly gather performance data.

6.2. TankSoar

TankSoar is a two-dimensional, tile-based implementation of the computer game genre known as a "first person shooter." The agent is a tank moving in a two-dimensional, tile-based maze. Fig. 7 shows a typical TankSoar map.

A tank can rotate in place to the left or right as well as move north, south, east or west. A tank that moves in a direction perpendicular to its bearing is effectively taking a side-step. A tank can also fire a missile forward that will damage an enemy tank if it hits it. A tank can also turn on and off its radar to better sense the world around it at the expense of energy. The radar's range can be set by the agent. A tank also has shields it can turn on to protect it against missiles, at the expense of energy. The agent can issue multiple simultaneous commands (e.g., move backwards and fire a missile), with the exception that it cannot move and turn at the same time. Many actions have parameters, so that there are between 200 and 700 unique actions available to the agent at any one time.

A tank has three resources. A tank's *energy* is used to power its radar and shields. An energy charger is placed randomly on the map at the beginning of the game that

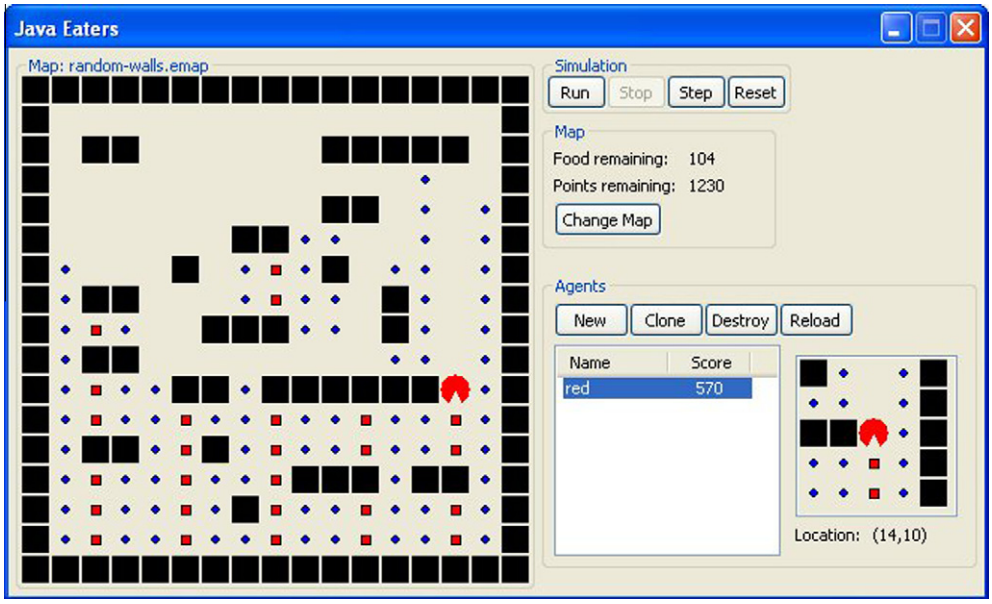

Fig. 6. The Eaters environment.

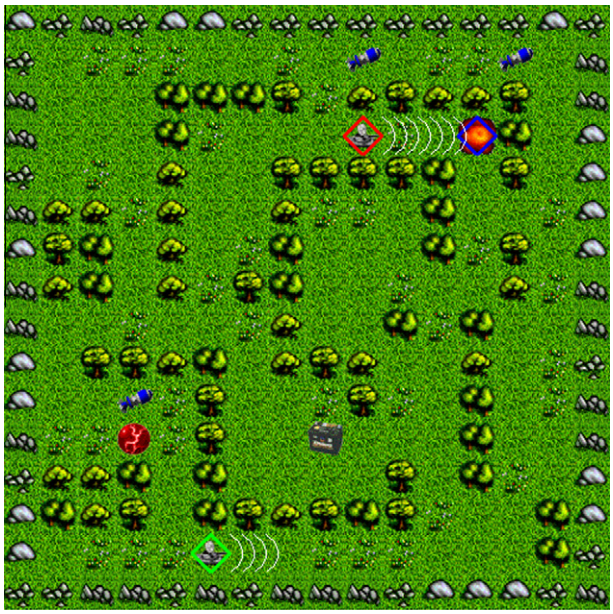

Fig. 7. The TankSoar environment.

the tank can use to recharge its energy. If a tank's *health* goes to zero, the tank is destroyed and then respawned at a random location on the map. Being hit with a missile lowers a 'tank's health. A health charger is placed randomly on the map at the beginning of the game that the tank can use to recharge its health. In order to fire, a tank must have *missiles*, which randomly spawn on the map during the game.

Success in the TankSoar domain is based on a score. Points are awarded for hitting other tanks with missiles and additional points are awarded if the enemy tank is destroyed.

TankSoar is more complex than Eaters. It is a dynamic environment where the tank has multiple actions, multiple sensors, and multiple opponents. It requires the agent not only be skilled in tactical combat, but also skilled at resource management and navigation. As a result, this environment allows us to test multiple cognitive capabilities.

7. Cognitive capabilities enabled by episodic memory

Each of the following sections presents an experiment in which we evaluate whether the inclusion of episodic memory in a cognitive architecture is sufficient to support the cognitive capabilities that we outlined earlier. The purpose of these experiments is to demonstrate that each cognitive capability can manifest with the aid of episodic memory. The data we present are not sufficient to prove that episodic memory supports the cognitive capability in the general case, but our implementations of the capabilities use general task-independent knowledge, so that there are no pre-existing barriers to general use of the capabilities. We also do not claim that episodic memory is the sole method for achieving these capabilities; however, in most cases, efficient access to a memory of prior situations is required, and episodic memory provides the necessary functionality and can be used for all of these capabilities. By realizing these cognitive capabilities with a task independent episodic memory we lay a foundation for future investigations into the generality of episodic memory's support for these cognitive capabilities.

For each cognitive capability, we describe a specific task where that the use of episodic memory supports the cognitive capability and in turn improves an agent's task performance. For each task, we present and discuss the results from this investigation, as well as any lessons learned.

7.1. Action modeling

Action modeling involves having the ability to predict how the environment will change when a given action is performed. This is valuable because it allows an agent to internally simulate alternative actions before committing to a specific one, so that it is essentially performing a look-ahead search. An agent can predict changes by using episodic memory to retrieve a previous experience where the agent attempted to apply the same action to a similar situation. To explore this cognitive capability, we created an Eaters agent that uses the following algorithm, which is illustrated in Fig. 8:

1. The agent's current state includes its current sensing and its current score. The state also contains elements related to the agent's reasoning. This state is represented by the box labeled (A) in the figure.
2. In any given state, the agent can move in at least two and as many as four different directions: north, south, east, or west. The agent determines which directions are available and proposes an action to move in each available direction. In the figure, the agent creates actions to move east, north or south. It is blocked by a wall to the west.
3. To decide which direction to move next, the agent evaluates each action using steps 4–7.
4. First, the agent creates an episodic memory cue that contains the agent's current state plus the action to be evaluated (e.g., move north). In other words, the agent is searching its episodic memory for a memory of taking the to-be-evaluated action in a similar situation. In the figure, the box labeled (B) represents the cue the agent would create to evaluate the action "move north."
5. Once the cue is created, the episodic memory system retrieves the episode that best matches this cue, recreating it in working memory. An example of the retrieved memory is labeled (C) in the figure. Note that this memory is not a perfect match but the recalled situation is similar enough because the agent is moving north to a cell that has the same content. In other situations, the memory may differ such that the agent makes a bad decision.

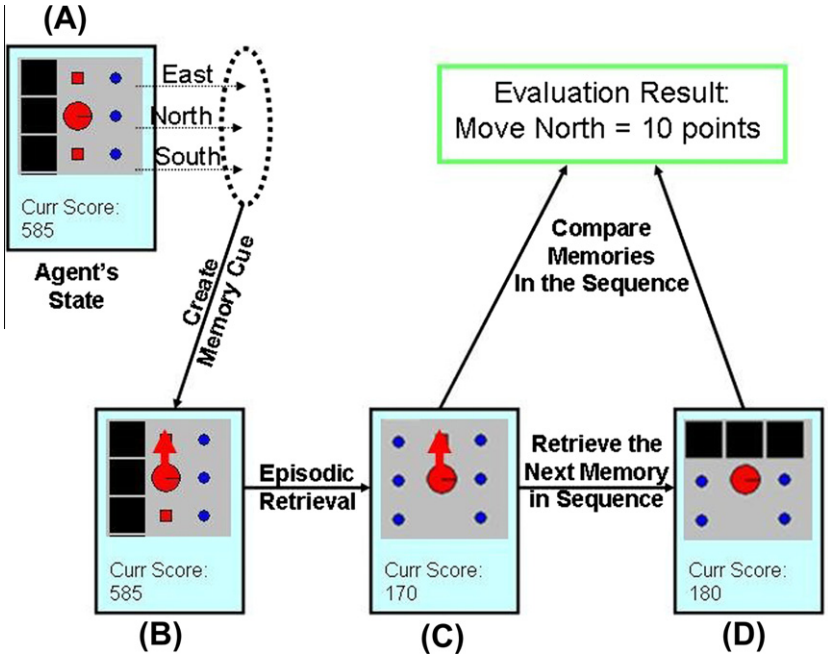

Fig. 8. An episodic memory Eaters Agent.

6. The agent records the score that it had in that episode and then requests the memory that occurred next in temporal order. The episodic memory system responds to this command and retrieves the immediate outcome of that situation and action. In the figure, this second retrieval is represented by the box labeled (D).
7. The agent then compares the scores from both memories and uses the difference between these two scores as a quantitative evaluation of the movement direction being considered.
8. Once all available actions have been evaluated in this manner, the agent selects the action with the highest evaluation. Ties are resolved randomly.

Aside from the goal of maximizing its score, the only knowledge given to this agent is an understanding of what actions it can take in the world (i.e., movement in a cardinal direction). The agent is not aware of the semantic meaning of those actions or the relative importance of the different information that is available from its senses.

Fig. 9 depicts the accuracy of this agent's action evaluations as it gains more and more episodic memories. This particular experiment was run five times. The results of each run were averaged and a mean smoothing with a windows size of ten was applied to achieve the final results that are shown in the figure.

The *x*-axis is the number of predictions that the agent has made so far and, thus, it is a measure of time. The *y*-axis indicates the fraction of the last ten predictions that were correct. For this experiment, a correct evaluation consists of predicting the correct change in score that will result for a given action. If the agent makes an incorrect prediction or is unable to make a prediction at all due to a failed episodic memory retrieval, then the outcome is considered a failure. The dotted line indicates the performance of an agent that makes random decisions.

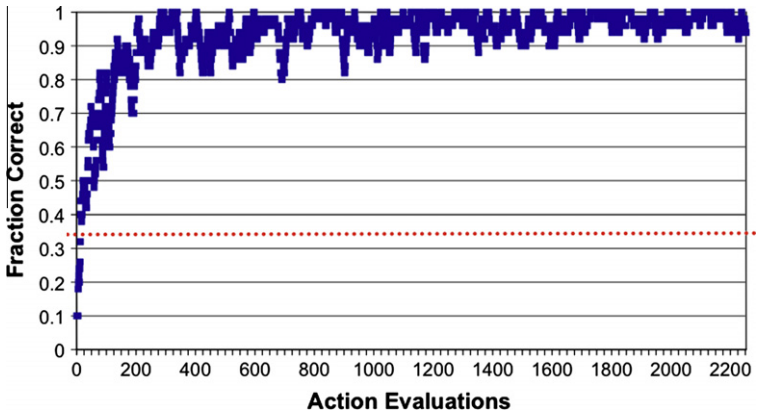

Fig. 9. Action modeling results (Eaters).

The agent begins with no episodic memories and so it is unable to retrieve a relevant memory and initially has a high failure rate. As the agent acts in the world, it builds up memories and its ability to model its actions improves rapidly, approaching perfect behavior. This demonstrates that episodic memory has the potential for being the basis for action modeling and is effective as such in Eaters. Addition examples of using episodic memory in Soar for action modeling in other domains can be found in Laird, Xu, and Wintermute (2010) and Xu and Laird (2010).

7.2. Decision-making based on past experiences

Action modeling allows an agent to predict the immediate outcome of an action. However, in many tasks, knowing the long-term outcome of an action is more critical to success than predicting the next state. In such situations, the cognitive capability of remembering the long-term success or failure that followed a particular action in a particular situation can lead to better behavior on the part of the agent. An agent can use its episodic memory to recall sequences of past states and actions to predict the outcome of a candidate action in the current state.

The TankSoar environment is well suited for demonstrating this cognitive capability. The outcome of a particular action is usually not immediate and often dependent upon future actions. For example, missiles fired by TankSoar agents take time to travel and their ultimate effect may be unknown for multiple time steps. As a second example, the decision to attack with low health and few missiles will almost certainly lead to disaster but that outcome may not occur immediately. As a result, success in the TankSoar domain requires the ability to predict the long-term success or failure that is likely to result from a particular action.

7.2.1. Heuristic control agent

As a basis for comparison, we started with an existing hand-coded agent for the TankSoar domain that is a competent player in that domain. This agent uses heuristics to select between four "modes" of action:

- **Attack**: Pursue and fire missiles at a visible enemy.
- **Chase**: Search for an enemy that the agent cannot see but can detect via other sensors.
- **Retreat**: This subgoal is selected when the agent detects a nearby enemy and does not have sufficient resources to attack.
- **Wander**: This subgoal is selected when the agent does not detect a nearby enemy.

7.2.2. An episodic learning TankSoar agent

To demonstrate the target cognitive capability, we created an episodic memory agent by further modifying the control agent. We removed the control agent's logic for selecting individual actions in Attack "mode." We replaced this logic with productions for selecting actions based upon episodic memory retrievals. This algorithm has the following steps and is illustrated by Fig. 10:

1. The agent first determines which actions are available based upon its current state. Certain actions may not be applicable because of the presence of obstacles or a lack of resources. In the figure, the box labeled (A) represents the agent's current state and the arrows emerging from the figure represent a list of possible actions. The agent uses episodic memory to evaluate each possible action as follows:
2. First, the agent creates an episodic memory cue that contains heuristically selected portions of the agent's current state plus the action to be evaluated. In other words, the agent is trying to retrieve an episode wherein it took the same action in a similar situation. In the figure, an example of the memory cue is represented by the box labeled (B).

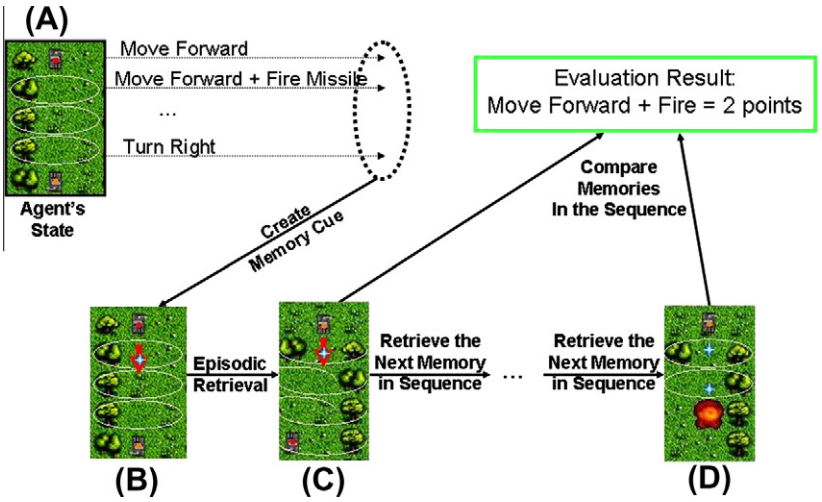

Fig. 10. An episodic memory TankSoar agent.

3. In the figure, the retrieved episodic retrieval is labeled (C). This episode is not an exact match to the cue, but it is the closest match that is available. Therefore, decisions based upon this retrieval may not be optimal.
4. The agent records the score that it had in that episode. Then, through repeated uses of a "next" command, the agent retrieves the sequence of subsequent episodes that occurred. It retrieves up to ten memories stopping if one of the following two events occurred in the most recently retrieved episode: (a) one of the two agents is destroyed or (b) the game ends.
5. The agent uses the overall change in score between the first and last retrieved episode as a quantitative evaluation of the action being considered. A discount factor is used such that actions that are more recent receive more credit for a particular outcome.
6. Once all the actions are evaluated, the agent selects the action with the highest evaluation. Ties are resolved randomly.

7.2.3. Results

Fig. 11 shows the episodic memory agent's performance against the control agent. The *x*-axis represents successive games. The agent retained its episodic memories from game to game and so had a larger episodic store at each successive game. The *y*-axis measures the agent's margin of victory: the control agent's score subtracted from the episodic memory agent's score. Thus, a negative margin of victory indicates a loss. The results depicted here are the average of ten repetitions of the same experiment. The error bars in the figure represent the range of 95% confidence.

As the graph shows, the agent's initial performance is poor but, through experience, the agent learns to outperform the hand-coded control agent by predicting the long-term outcome of its actions. By analyzing the agent's behavior, we identified three tactics that emerged that had a significant impact on its performance. First, the agent learned to dodge short-range enemy missile attacks before they hit. Second, the agent learned to back away from an enemy while firing its missiles. This delayed the impact of enemy missiles and opened up future opportunities to subsequently dodge. Finally, the agent learned to move out of the enemy's sight when it was in a tactically unfavorable situation.

All three of these learned behaviors stemmed from the fact that reliable long-term results occurred when the agent took the same action in many similar states. In situations where the outcome of an action was less reliable, the agent's behavior was more variable. Given these observations, we expect that improvements in behavior in other domains would rely upon consistent experiences in terms of success and failure.

7.3. Retroactive learning

Retroactive learning is the ability to relive an experience when more resources are available in order to learn things from those experiences that could not be learned when they occurred. For example, an agent might gain new knowledge that would have improved the way it learned in a previous experience. Alternatively, an agent may not have time to apply a complex learning algorithm during an experience but can relive the experience in order to learn from it when more time is available. An agent with an episodic memory can review its memories to relive its past for this purpose.

We demonstrate this cognitive capability in the Eaters domain using the following high-level algorithm:

1. The agent acts randomly for a fixed period of time gathering episodic memories.
2. After this time, the agent stops and retrieves memories by constructing cues consisting of the agent in all possible combinations of neighboring cells.
3. Based upon its retrievals, the agent used Soar's existing rule-learning mechanism (Laird, Rosenbloom, & Newell, 1986) to create new productions that dictate which action to take based upon the contents of neighboring cells.

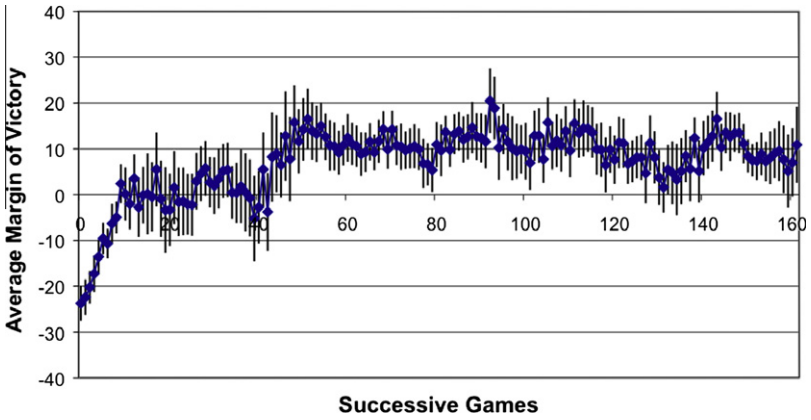

Fig. 11. Subsequent results with learning from past success and failure.

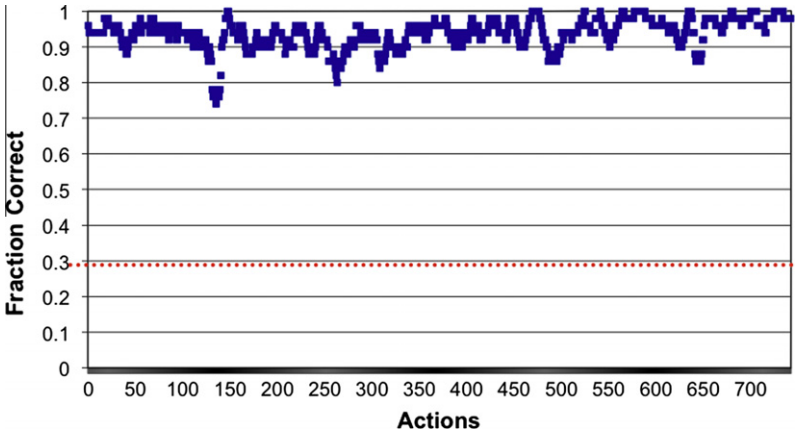

Fig. 12. Retroactive learning results.

4. After this period of reflection, the agent resumes its movement through the maze using its new knowledge.

Fig. 12 depicts the performance of this retroactive learning agent in the Eaters environment during step 4. The *y*-axis measures the fraction of the last 10 actions that were correct while the *x*-axis represents successive actions in the world. The data is an average of five runs of the agent. The dashed line indicates the expected performance of an agent that selects randomly among available actions. Before retroactive learning was applied, the agent performed at this level.

This graph indicates that the retroactive Eater was able to learn near-ideal behavior via retroactive learning. The situations where the agent's behavior was imperfect after retroactive learning were due to encountering novel situations that had no analog in the episodic store. An agent with a sufficiently broad experience would have learned the correct action for each situation. More generally, we would expect the agent to perform similarly at any task that has a relatively small number of states and a consistent result from each action taken in each state.

7.4. Boosting other learning mechanisms

Learning mechanisms can be categorized into "lazy" and "eager" learners. An eager learner will distill information immediately into knowledge that can be used to select future action. A lazy learner will collect experiences but not learn from them until they are relevant to selecting an action in a particular situation. The potential exists for the knowledge gathered by a lazy learner to become input for an eager learner. The lazy learner still delays learning until it is needed, but uses an eager learning mechanism to learn from its knowledge. Thus, in future situations, the eagerly learned knowledge can be applied without delay.

Episodic learning is a form of lazy learning. The episodes stored in the memory are raw knowledge that is not used to make decisions until the agent needs them. When learning takes place using the knowledge in this episodic store, an eager learner can be applied to improve the speed at which an agent arrives at future decisions.

To demonstrate an episodic memory system's ability to boost another learning mechanism we combined our implementation of episodic memory with Soar's built-in learning mechanism: chunking (Laird et al., 1986). Chunking allows a Soar agent to cache deliberate processing and turn it into reactive processing. As a result, once an agent has made a decision, it will make the same decision when the same circumstances arise in the future.

We modified an agent for the Eaters domain to using chunking to store the results of decisions made with episodic memory. In other words, these chunks compile the processing that generated the episodic memory retrievals so that the agent can skip these retrievals in the future. Since the episodic memory agent makes many decisions, chunking eliminates the number of episodic retrievals over time and, as a result, allows the agent to eat faster.

Fig. 13 depicts a comparison of an episodic memory agent with chunking and the same agent without the chunking mechanism. For comparison, the figure also includes results from an agent that requires only one decision cycle per action and always takes the correct action. In this case, we defined a correct action as ideal behavior for a greedy agent. The *y*-axis measures each agent's score while the *x*-axis represents successive Soar decision cycles instead of successive actions as in previous results. This means that the graph is weighing both the effectiveness of the agent's actions and the speed at which the agent makes decisions. The data shown are an average of five runs of the respective agents, which provide a statistically significant result.

As the figure depicts, the addition of chunking significantly improves the agent's performance. This improvement in behavior results because both chunking and episodic memory are present. To be effective in the general case this technique relies upon decisions based upon the episodic memory to be correct ones. A more stochastic environment or a less certain situation can cause the eager learner to learn incorrect knowledge from the episodic store.

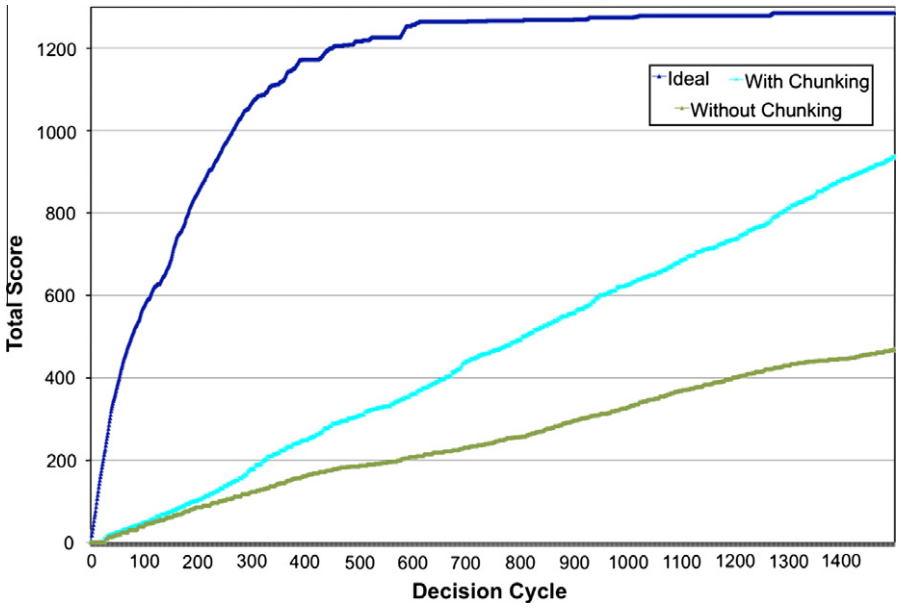

Fig. 13. Boosting other learning mechanisms results.

7.5. Virtual sensing

When an agent originally senses something, it may be irrelevant to its current task. Then, at some future point, that past sensing may become important. An agent with episodic memory has a record of its past sensing embedded in its recorded episodes. When necessary, the agent can retrieve these episodes to relive or remember the sensing.

To demonstrate this cognitive capability in the Tank-Soar domain, we chose the task of locating the battery used to recharge the agent's energy supply. When the tank's energy supply runs low, the agent is effectively blinded since it can no longer use its radar. While an agent could deliberately construct a map while it has energy, we wanted to demonstrate that an agent with an episodic memory could construct a map by relying on its episodic memory.

Our agent uses the following algorithm:

1. The agent moves forward or backward until it is blocked.
2. If the agent already has knowledge of a path from this position to the battery, then it turns to the direction specified by that path and returns to step 1. For this agent a "path" is a location (x, y coordinate) combined with a direction (north, south, east or west). If the agent does not know of a path to the battery from this location, it proceeds to step 3.
3. The agent attempts to retrieve an episodic memory of seeing the battery from this position. The cue for this retrieval consists of the agent's current location and the perceptual elements that represent seeing the battery on its radar. If the retrieval is unsuccessful, the agent proceeds to step 4. If the retrieval is successful, the agent records a path in this position and moves directly toward the battery. It then resets to step 1. If the retrieval fails, it proceeds to step 4.
4. The agent attempts to retrieve an episodic memory of being in its current position and seeing another location for which it has already recorded a path to the battery. If the retrieval is successful, it creates a new path in working memory that directs the agent from this position toward the origin of the existing path and resets to step 1.
5. If step 4 fails, the agent moves in a random direction and resets to step 1.

Over time, an agent using this algorithm constructs a set of paths that direct it from any position in the maze to the battery. The learned paths, while effective, are not necessarily optimal.

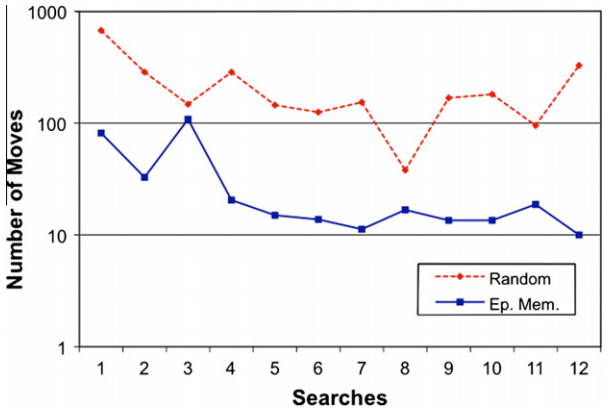

Fig. 14. Virtual sensors results.

Fig. 14 depicts the results from this experiment. The *y*-axis has a logarithmic scale and measures the number of moves required to reach the battery. Thus, a lower score is better in this figure. The *x*-axis represents subsequent searches over time. For the first search, the agent has only a few episodic memories. Over time, the agent learns more paths and acquires more episodic memories.

As the figure depicts, the time required to find the battery diminishes as the agent acquires more episodic memories. Specifically, these data show that the agent is able to find the target an order of magnitude faster than a random search. These results demonstrate that it is possible for an agent to use its episodic memory to virtually sense things in the environment when it cannot actually see them. The effectiveness of this approach in the general case relies upon the agent being able to reconstruct the knowledge it desires from the given episodes. Decisions about the content of episodic memories or even the structure of how they are stored could prevent effective virtual sensing in other environments.

8. Conclusions and future work

In this paper, we presented a broad picture of the challenges and benefits of providing an episodic memory to an intelligent agent. We then cataloged some of the cognitive capabilities episodic memory might be able to support in an agent. Next, we described an implementation of a general-purpose episodic memory for the Soar architecture. Finally, we presented the results from using our episodic memory architecture's ability to facilitate these cognitive capabilities. By investigating these capabilities and demonstrating the possibility of supporting them in an artificially intelligent agent, we have established the possibility that a single, general memory system can provide them.

This research presents demonstrations of five cognitive capabilities: virtual sensing, action modeling, decision-making based on past experiences, retroactive learning and boosting other learning mechanisms. In addition to those cognitive capabilities, we have identified the following additional cognitive capabilities that intuitively appear to require an episodic memory. Given our goal to create a comprehensive episodic memory, our future work will focus on demonstrating these additional capabilities with our system.

- *Noticing significant input*: One challenge for an agent is determining which aspects of its current situation are most important, with changes to the environment being an important indicator of relevance. Episodic memory allows an agent to detect changes by comparing the current situation to prior memories.
- *Detecting repetition*: Given the limited memory of most AI agents, it is difficult for them to detect when they repeat the same sequence of actions without making any progress on their current task. Episodic memory provides the necessary memory to detect when the same situation is encountered, or the same action is tried repeatedly.
- *Environment modeling*: In many domains, the environment has its own dynamics (e.g., Sunset has been around 6:30 pm lately.). An episodic memory provides a record of these changes and, thus, allows the agent to predict them in similar situations in the future.
- *Managing long term goals*: An agent with multiple goals must often switch between them because of environmental demands and opportunities. This requires that the agent be able to record the progress it has made for a given goal and restart or recover its progress. Furthermore, a goal must sometimes be suspended for an indefinite period of time. An agent can create a prospective episodic memory (Kliegel et al., 2007), i.e., it can remember to reinstantiate a suspended goal in response to a future expected event. To schedule these goals, an agent needs to recall goals it has suspended and remember the progress it has made toward each one.
- *Sense of identity*: An agent with a sense of identity potentially gains a greater ability to recognize its own behavior and analyze it compared to the behavior of other agents. For humans, one's sense of identity is rooted in memories of past experiences, which indicates that episodic memory has an important role to play in this capability.
- *Reanalysis given new knowledge*: When a learning agent receives new knowledge about its environment, inferences and behavior it has learned in the past may no longer be valid. An episodic memory allows an agent to review its experiences that relate to the new knowledge and change its behavior accordingly.
- *Explaining behavior*: The ability to remember what you did in the past allows you to explain your actions to others and allow them to instruct you or you to instruct them (e.g., Why did you go left instead of right?). An agent can use its episodic memory to recall the situation in question as well as the decisions it made in that situation.

We assess the strengths and weaknesses of our approach to creating a task independent episodic memory as follows.

Integrating our memory system into a cognitive architecture means any agent constructed with that architecture automatically gains the capabilities granted by that episodic memory. Furthermore, the episodic memory operates within the restrictions defined by a body of existing research aimed at creating general intelligence.

By carefully examining the design decisions that were made in constructing our episodic memory system, we implicitly define a set of possible, alternative implementations. This allows us to methodically compare specific implementations and select the ones that are most effective. However, this space of possible implementations is by no means complete. Design decisions that we overlooked will hide portions of the space from our investigation.

By defining, in advance, a set of cognitive capabilities that could be facilitated by an episodic memory, we provide a metric for measuring the quality of an episodic memory architecture. These capabilities also provide a clear direction for future research. However, due to their introspective nature, any set of cognitive capabilities we define is inherently imperfect and likely incomplete. Furthermore, the qualitative nature of some of these cognitive capabilities makes them difficult to measure.

Finally, by building a complete episodic memory system and refining it we move expediently to a system that can be used for various research tasks. As a result, we rapidly gain experience and insight about the abilities and limitations of episodic memory. However, by committing to a single approach it is more difficult to gain perspective on the best method for building an episodic memory architecture.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 0413013. We would also like to acknowledge and express our gratitude to Nate Derbinsky, whose work in this area has brought considerable clarity to our conclusions.

References

Anderson, J. R. (2007). *How can the human mind occur in the physical universe?* New York, NY: Oxford University Press.

Atkeson, C., Moore, A., & Schaal, S. (1997). Locally weighted learning. *Artificial Intelligence Review, 11*(1–5), 11–73.

Derbinsky, N., & Laird, J. E. (2009). Efficiently implementing episodic memory. In *Proceedings of the 8th international conference on case-based reasoning*.

Goodman, M. (1993). Projective visualization: Acting from experience. In *Proceedings of the eleventh national conference on artificial intelligence,* 54–59. San Jose, CA: AAAI Press.

Ho, W. C., Dautenhahn, K., & Nehaniv, C. L. (2003). Comparing different control architectures for autobiographic agents in static virtual environments. *Intelligent virtual agents, Springer lecture notes in artificial intelligence, 2792*, 182–191.

Jones, R., Tambe, M., Laird, J., & Rosenbloom, P. (1993). Intelligent automated agents for flight training simulators. In *Proceedings of the third conference on computer generated forces and behavioral representation*. University of Central Florida. IST-TR-93-07.

Kieras, D., & Meyer, D. E. (1997). An overview of the EPIC architecture for cognition and performance with application to human–computer interaction. *Human–Computer Interaction, 12*, 391–438.

Kliegel, M., McDaniel, M. A., & Einstein, G. O. (2007). *Prospective memory*. Psychology Press.

Kolodner, J. (1993). *Case-based reasoning*. San Mateo, CA: Morgan Kaufmann Publishers.

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

Laird, J. E. (2008). *Extending the Soar cognitive architecture*. Memphis, TN: Artificial General Intelligence Conference.

Laird, J. E., Xu, J. Z., & Wintermute, S. (2010). Using diverse cognitive mechanisms for action modeling. In *Proceedings of the tenth international conference on cognitive modeling.* Philadelphia, PA.

Langley, P. (2006). Cognitive architectures and general intelligent systems. *AI Magazine, 27*, 33–44.

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

Nolan, C. (Director) (2000). Memento [Film]. Newmarket Films. http://us.imdb.com/title/tt0209144/.

Nuxoll, A., & Laird, J. (2004). A cognitive model of episodic memory integrated with a general cognitive architecture. In *Proceedings of the international conference on cognitive modeling*.

Nuxoll, A., Laird, J., & James, M. (2004). Comprehensive working memory activation in Soar. *Proceedings of the International Conference on Cognitive Modeling*, 226–230.

Nuxoll, A., Tecuci, D., Ho, W. C., & Wang, N. (2010). Comparing forgetting algorithms for artificial episodic memory systems. In *Proceedings of the thirty sixth annual convention of the society for the study of artificial intelligence and simulation of behaviour (AISB)*. UK: Leicester.

Ram, A., & Santamaría, J. C. (1997). Continuous case-based reasoning. *Artificial Intelligence, 90*(1-2), 25–77.

Schank, R. (1999). *Dynamic memory revisited*. Cambridge University Press.

Sheppard, J. W., & Salzberg, S. L. (1997). A teaching strategy for memory-based control. *Artificial Intelligence Review, 11*(1-5), 343–370.

Sun, R. (2006). The CLARION cognitive architecture: Extending cognitive modeling to social simulation. In Ron Sun (Ed.), *Cognition and multi-agent interaction*. New York: Cambridge University Press.

Tecuci, D., & Porter, B. (2007). A generic memory module for events. In *Proceedings to the 20th Florida artificial intelligence research society conference (FLAIRS20)* (pp. 152–157).

Tulving, E. (1983). *Elements of episodic memory*. Oxford: Clarendon Press.

Tulving, E. (2002). Episodic memory: From mind to brain. *Annual Review of Psychology, 53*, 1–25.

Vere, S., & Bickmore, T. (1990). A basic agent. *Computational Intelligence, 6*, 41–60.

Xu, J. Z., & Laird, J. E. (2010). Instance-based online learning of deterministic relational action models. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*. Atlanta, GA.