Extending Cognitive Architecture with Episodic Memory

Andrew M. Nuxoll and John E. Laird

University of Michigan
2260 Hayward St.
Ann Arbor, MI 48109-2121
{anuxoll, laird}@umich.edu

Abstract

In this paper, we explore the hypothesis that episodic memory is a critical component for cognitive architectures that support general intelligence. Episodic memory overlaps with case-based reasoning (CBR) and can be seen as a task-independent, architectural approach to CBR. We define the design space for episodic memory systems and the criteria any implementation must meet to be useful in a cognitive architecture. We present an implementation and demonstrate how episodic memory, combined with other components of a cognitive architecture, supports a wealth of cognitive capabilities that are difficult to attain without it.

Introduction

Episodic memory was first described in detail by Tulving (1983). Episodic memory is what you "remember," and includes contextualized information about specific events, such as a memory of a vacation or the last time you had dinner. In contrast, semantic memory is what you "know," and consists of isolated facts that are decontextualized – they are not organized in a specific experience and are useful in reasoning about general properties of the world, such as knowing that George W. Bush is president in 2007. Episodic memories allow you to extract information and regularities that where not noticed during the original experience and combine them with current knowledge. Episodic memories can sometimes be retrieved as a sequence (like a movie) and commonly contain approximate or relative temporal information.

Episodic memory is a capability that we take for granted in humans, except when an accident or disease disables it. When that happens, the resulting amnesia is devastating. Oddly enough, the vast majority of integrated intelligent systems ignore episodic memory, which often dooms them to what can be achieved by people with amnesia, which is demonstrably limited. Our hypothesis, supported by human amnesia data, is that episodic memory is critical for providing a memory of previous events, but also for supporting a host of additional cognitive capabilities that greatly enhance the reasoning and learning capabilities of an integrated intelligent agent.

There are only limited examples of computational implementations of episodic memory for integrated agents. The "basic agent" created by Vere and Bickmore (1990) had a primitive episodic memory capability (a linked list of prior states), that was used only in a limited capacity (such as to answer questions). Ho et al. (2003) describe an agent that uses a task-specific implementation of episodic memory to locate previously encountered resources.

Episodic memory is related to case-based reasoning (CBR) (Kolodner 1993). In many CBR systems, a case describes the solution to a previously encountered problem that the system retrieves and adapts to new problems. No matter what the exact approach, the structure of cases (the specific fields in the case) are designed by a human for a specific task or set of tasks, limiting their generality. For example, continuous case-based reasoning (Ram & Santamaría 1997) relies upon cases that consist of the agent's sensory experiences, but none of its internally generated abstractions.

We extend the CBR paradigm by integrating episodic memory with a general cognitive architecture and developing task independent mechanisms for encoding, storing, and retrieving episodes, none of which make assumptions about the structure or contents of the episodes.

In this paper, we present the design space for episodic memory implementations, followed by a description of our implementation(s) of episodic memory as integrated with the Soar cognitive architecture. Our hypothesis is that episodic memory supports multiple cognitive capabilities, which we describe. We confirm our hypothesis with implementations of three of those capabilities in a moderately complex task. Previous work on integrating episodic memory with Soar (Nuxoll & Laird, 2004) has studied only a simple task and a single cognitive capability.

Episodic Memory Design Space

Learning mechanisms embedded in cognitive architectures have three stages: *encoding* information from the current situation, *storing* it, and *retrieving* it. There is also a final stage of *using* the information to influence behavior, but this draws on the general capabilities of the architecture, independent of the structure of episodic memory. Each of these stages has subparts, with design choices that together determine the space of possible episodic memory designs. Thus, one aspect of our research is to search through this space of possible designs considering the implications of

each choice on episodic memory functionality. This section describes the design space and the choices we made for our current implementation. For the most part, we have chosen the most general approach to episodic memory.

Encoding

- **Encoding initiation**: When is an episode encoded? This can be determined deliberately by the agent or could be automatic at some specific time during the agent's processing. In our current approach, an episode is encoded each time the agent takes an action.
- **Episode determination**: What are the contents of an episode? In our implementation, an episode consists of the agent's current state, which includes perception, motor commands, and internal data structures. When the episode is recorded, only features whose activation level (defined later) exceeds a preset threshold are included.
- **Feature selection**: Is a subset of features in the stored episodes selected for matching during retrieval? In our current implementation, the entire memory is used.

Storage

- **Episode structure**: What is the structure of the episodic memory store? This is important because the structure influences the efficiency of storage and retrieval. We describe two alternatives in subsequent sections.
- **Episode dynamics**: Does the content or organization of the episodic store change over time, such as through forgetting or generalization? In our system, the episodic store is static without forgetting or reorganization.

Retrieval

- **Retrieval initiation**: When is an episode retrieved? Is retrieval initiated deliberately or is spontaneous retrieval possible using the complete current situation as a cue? In our implementation, deliberation retrieval is initiated when the agent creates a cue.
- **Cue determination**: How is the cue specified? In our implementation, a cue is created in a reserved portion of the agent's temporary memory and the cue can include not only specifications of what exists in the retrieved episode, but also what cannot exist.
- **Retrieval:** Given a cue, which episode is retrieved? Our system uses activation and recency biased partial match. Details of our matching algorithm are in the next section.
- **Retrieved episode representation**: When an episode is retrieved, how is it represented in the agent? Our implementation represents the episode in its entirety with an annotation that it is a retrieved episode.
- **Retrieval meta-data**: Is there additional meta-data about the episode and its match to the cue? Our system's meta-data includes: data about the strength of the match and the relative time that the episode was recorded.

Outside of the mechanics of the episodic memory system is the question, "How is the retrieved memory used?" Addressing this question leads directly to the cognitive capabilities described later in this paper.

One final concern is that an episodic memory system must meet practical resource requirements – the cost of using it should not outweigh its benefit. Thus, a goal is that the growing need for computational resources can be met at reasonable cost for the predicted existence of the agent.

Episodic Memory Implementation in Soar

Our implementation of episodic memory is embedded in Soar (Newell 1990) as shown in Figure 1. Soar represents it procedural long-term knowledge as production rules. It represents short-term declarative knowledge in its working memory, which includes internally generated structures, motor commands, and structures created by perception. When conditions of rules match working memory, rules "fire" and create new structures in working memory. Deliberate action in Soar is generated by rules proposing, comparing, and evaluating, and applying operators. While, rules fire in parallel as soon as they match, only a single operator is selected and fired during Soar's decision cycle.

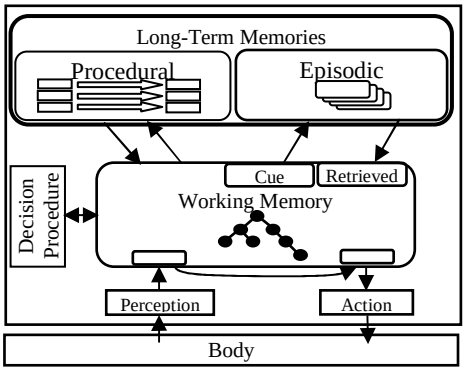

Figure 1: The Soar Architecture with Episodic Memory

To provide a task-independent method of identifying important working memory elements (WMEs), a working memory activation system was added to Soar (Nuxoll, et al. 2004) that was based on the activation scheme in ACT-R (Anderson and Labiere 1998). The activation levels of WMEs are increased whenever it is tested by a rule, indicating it is important to the current processing. The activation of a WME decays exponentially over time.

Episodic Memory Integration

In our implementation, snapshots of working memory are captured and automatically stored in episodic memory. Everything in working memory is stored in the episode except for WMEs with very low activation (indicating they were not recently created or tested).

Episodic memory is queried by an operator creating a structure in the cue area of working memory. The episode that is the best match to the cue is found and deposited in the retrieved area of working memory.

Episodic Memory Matching Algorithms

An effective and efficient episodic memory system is dependent on the data structures for storing episodes and the algorithms for retrieving episodes, which must minimize storage and computational expense, while providing an expressive language for retrieving episodes using partial match. (Early experiments using exact match, while efficient proved inadequate for flexible retrieval.) Thus we developed and experimented with two algorithms and associated data structures, which we label instance-based (where each episode instance is explicitly stored) and interval-based (where episodes are represented by a record of the intervals when individual WMEs existed). Both partial matching algorithms rely upon a data structure called the working memory tree. The tree retains a record of each unique WME that has ever existed.

The instance-based approach uses a data structure that holds instances of each episode. Instead of storing the actual WMEs, it stores pointers to nodes in the working memory tree (see Figure 2). A memory cue is a conjunction of WMEs. In response to a cue, the instance-based algorithm retrieves a matching episode as follows:

1. For each element in the cue, the corresponding entry in the working memory tree is found, which includes a list of all episodes it appeared in.
2. The set of episodes that contain at least one feature of the cue are collected.
3. The "best matching" episode is then selected. In the simple case, this means selecting the episode that includes the most cue elements (biased by recency).

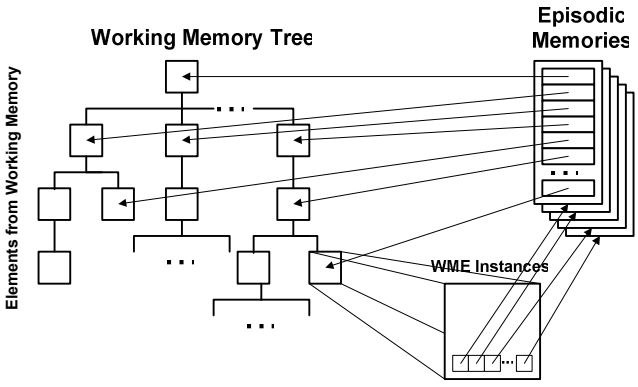

Figure 2: Data structures for instance-based matching

The instance-based algorithm requires O(nm) time to complete (where 'n' is the size of the cue and 'm' is the number of episodic memories that include at least one cue element). The cue is usually small and does not grow over time. In the worst case, the size of 'm' is equal to the size of the episodic memory store, but its expected size is much smaller, especially as different tasks are pursued. Empirical measurements of existing systems show that the time grows linearly with the size of the episodic store.

Our second matching algorithm was inspired by the observation that two sequential episodes are very similar. In this approach, we eliminate the explicit representation of the episodic memories. To compensate, we modified the working memory tree so that each node contains a list of the intervals in which the WME existed (see Figure 3).

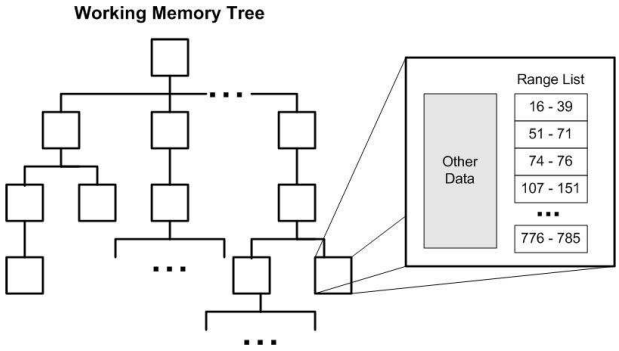

Figure 3: Data structures for interval-based matching

When a memory cue is created, the interval-based algorithm retrieves a matching episode as follows:

1. For each entry in the memory cue, the system finds the corresponding node in the working memory tree, and extracts its interval list.
2. All extracted intervals are merged together into a larger set of intervals where each interval corresponds to a unique combination of matching cue elements at a specific time.
3. The merged list is traversed to locate the range with the highest match score.
4. The episode is recreated by traversing the working memory tree and locating each node that includes the selected interval.

The complexity of this algorithm is $O(n^2l)$ where 'n' is the size of the cue and 'l' is the average size of the list of intervals in each node of the working memory tree. As with the instance-based algorithm, these two variables are not independent. If the size of the cue is small and relatively constant, we observe linear growth in processing time. In our research domains, the interval-based algorithm was approximately 15% faster than the instance-based algorithm and required roughly on quarter of the memory to store the same number of episodes.

Both of these methods fail the computation bound requirements. There are three responses to this problem:

1. Develop new algorithms that meet these requirements. Given the inherent need for more and more memory to store episodes, this is unlikely using standard von Neumann architectures, but may be possible with alternative computational architectures, such as content-addressable memories.
2. Modify the dynamics of episodic memory so that the number of episodes is bounded. This could be achieved via forgetting. Although this has possibilities, any fixed bound will ultimately have a negative impact on a general intelligent agent.
3. Determine the bound on the number of episodes that can be efficiently processed and restrict our use of episodic memory to problems that meet that limit.

We have adopted approach #3. We use and experiment with episodic memory in agents we develop, but we are

limited to the length of our system's experiences by the computational demands of episodic memory (these correspond to roughly four hours of human experience). In the end, option #1 is probably the only viable approach, but that requires development of special purpose hardware, which is beyond our current resources.

Selection Bias

As episodic memory grows, usually there will be more and more episodes that have some relationship to the cue. Of all these episodes, which one should be selected? The simplest approach it to choose the episode that includes the largest number of elements from the cue, giving all elements of the cue equal weight. This puts the onus on the agent to create a selective cue, which can be difficult. The agent might not know which features are important. Early work on episodic memory in Soar demonstrated that using activation level for biasing episodic retrieval significantly improved the quality of performance (Nuxoll & Laird 2004). Activation provides a simple but powerful heuristic for selecting which elements of the cue are important within the current problem-solving context.

Our research has extended that work by enriching the cue description so that the agent can also specify specific WMEs that *cannot* be in the cue. This extension improves performance by eliminating the retrieval of episodes that the agent knows it will not use. In addition, when requesting a retrieval, the agent can signal that a specific episode should not be retrieved (such as when an episode has been retrieved and found to be inadequate).

Episodic Memory Cognitive Capabilities

Once we have an episodic memory system, what good is it? In this section, we list cognitive capabilities that episodic memory helps support. We do not claim that this list is complete, or that episodic memory is *necessary* to achieve these capabilities (which would require a much more extensive analysis of the cognitive capabilities and alternative implementations). We are prepared to claim and demonstrate that a single implementation of episodic memory is *sufficient* for supporting some of these capabilities.

Sensing:

- Noticing Novel Situations: By failing to retrieve episodic memory similar to the current situation, the agent can detect when it is in a novel situation. "I've never been here before."
- Detecting Repetition: Retrieval of episodes that are identical (or close to identical) can inform an agent that it is repeating situations and possibly not making any progress toward its goals. "I've was just here. I must be going in circles."
- Virtual Sensing: An agent can retrieve past episodes that include sensory information relevant to the current task that are beyond its current perceptual range. "I remember seeing a coffee shop just around the corner from here."

Reasoning:

- Action Modeling: An agent can retrieve an episode of a similar situation where it has performed an action. It can then compare that episode to what came next to determine how the action affects the world.
- Environment Modeling: This is similar to action modeling, except the agent is using sequences of episodes to predict how the world will change (independent of its own actions).
- Predicting Successes/Failures: This extends the previous two by using episodic memory to recall the expected value of an action and the ensuing changes in the world that eventually lead to some feedback for the agent in terms of success or failure.
- Managing Long Term Goals: Episodic memory remembers whether or not goals have been achieved, eliminating the need to maintain goals in working memory, which for long-term goals can be difficult.

Learning:

- Retroactive Learning: An agent might acquire experiences when performing under time pressure. Later, when sufficient time becomes available, the agent can analyze the experience.
- Reanalysis of Knowledge: As new information becomes available, episodes can be retrieved and reanalyzed using the new information. "Now that I know he was lying, everything that happened makes sense."
- Explaining Behavior: An agent can use episodic memory to replay earlier behavior to explain its behavior to others, tell stories or relate prior experiences.
- "Boost" other Learning Mechanisms: Episodic memory can provide the "grist" for the "mill" of other learning mechanisms such as explanation-based learning and reinforcement learning.

These capabilities could be realized by individual architectural modules. However, many, if not all of them require the recording and retrieval of the agent's experiences. A single general-purpose episodic memory simplifies the implementation of the cognitive capabilities and eliminates redundant functionality.

Experimental Environment

For the experiments described in this paper, we use a single environment called TankSoar. In TankSoar, the agent controls a tank moving in a two dimensional (tile-based) maze. It has three resources: missiles, energy, and health, which are expended by the tank's actions in the world. Without energy, a tank cannot see or shield itself against attach. Without missiles, a tank cannot attack another tank, and without health, a tank is destroyed.

The agent's goal is to destroy other tanks in the maze by firing missiles while avoiding enemy missiles. Figure 4 depicts a portion of a TankSoar map. While the human observer can see the entire map, the tank has limited sensors, with radar being the most important. A battery and health charger recharge the agent's energy and health.

Figure 4: A subset of the TankSoar environment

For each of the cognitive capabilities we implemented, we focused on a specific task that a TankSoar agent might undertake. For these experiments, we used the instance-based algorithm (biased by activation and recency) because even though it is less efficient, it is easier to modify.

Cognitive Capability: Action Modeling

An agent with episodic memory can predict the immediate effects of its actions by examining similar situations in its past wherein it took the same actions. Our action-modeling experiment in TankSoar focuses on the problem of energy management. A tank uses its radar to sense the environment immediately in front of it. The radar can be set to different distances with further distances requiring more energy. Energy is wasted if the radar is blocked by an obstacle (e.g., a wall or another tank).

Our agent uses its episodic memory to predict what it will see when it turns on its radar, and uses that information to set the radar distance. In this task, it is essential that the episodic memory is effective at retrieving a relevant memory. The agent performed best when an exact match (even with lower activation) was preferred over the best activation-biased match (and this preference was used in all of the remaining experiments).

Figure 5 depicts the agent's performance over one hundred radar settings while the agent explores a map. The y axis is the fraction of the last ten settings that were correct (the first nine settings are not shown). Each data point is the average of five runs. The dashed line at the bottom of the graph indicates the performance of an agent that selects its radar setting randomly. As the graph shows, the agent quickly learns to make effective radar settings as it navigates the maze.

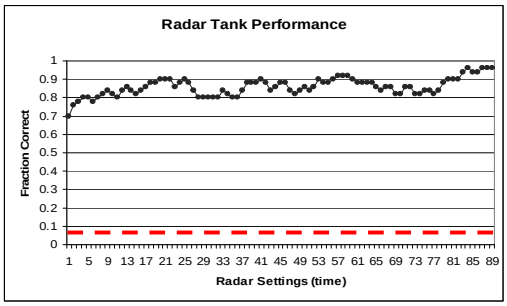

Figure 5: Agent performance in the action modeling task.

Cognitive Capability: Virtual Sensors

When an agent senses something at one moment, it may seem irrelevant to its task. Then, at some future point, that past sensing may become important. An agent with episodic memory can retrieve details of its sensing. This capability is most useful in environments with large bodies of data that are irrelevant to the current task, but may be relevant to future tasks.

To demonstrate this cognitive capability, we chose the task of locating the battery used to recharge the tank's energy supply, where we assume the tank does not know the importance of the battery until it is low on energy. When the tank's energy supply runs low the tank remembers where it has last seen the battery and uses that information to direct its search for the battery.

Figure 6 depicts the number of moves required to find the battery for two agents over twelve subsequent searches. The first agent searches randomly for the battery. The second agent uses its episodic memory to attempt to find the battery before resorting to random search.

As the graph shows, the virtual sensor agent located the battery an order of magnitude faster than the random agent. In addition, the episodic memory agent's performance continued to improve as it gained more memories.

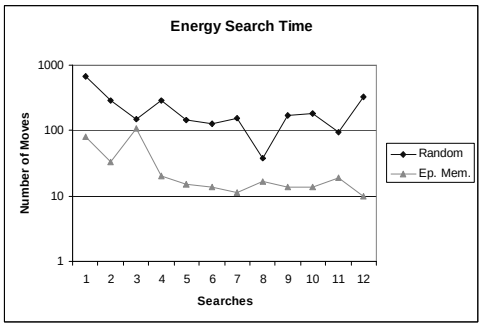

Figure 6: Agent performance in the virtual sensors task (Note: the ordinal is in logarithmic scale.)

Cognitive Capability: Learning from Past Successes and Failures

Action modeling allows an agent to predict the immediate outcome of an action. However, success in many tasks requires a coherent strategy with multiple actions taken in concert. To demonstrate this cognitive capability, we took an existing hand-coded agent for the TankSoar domain, removed its tactical knowledge and modified it to use episodic memory to make tactical decisions. When considering a particular action, the episodic memory tank would query episodic memory to find an episode in which it took the same action in a similar, but possibly not identical, situation. It would then evaluate the overall effectiveness of each of those actions by doing repeated retrievals of subsequent episodes until there was a clear outcome or a maximum depth was reached. To account for the delay between the agent's

action and success or failure, we added a discount factor. The agent did some tracking of the effectiveness of its episodes. If a memory consistently led to poor decisions, the agent would avoid using that memory in the future. Each episode contained ~120 WMEs.

We pitted the episodic memory tank against the original tank. Figure 7 shows how the episodic agent's average margin of victory changes as more games are played – a game ends when one tank achieves 50 points. At first, the episodic memory tank loses most games (negative margin of victory) but after 20 matches it wins consistently.

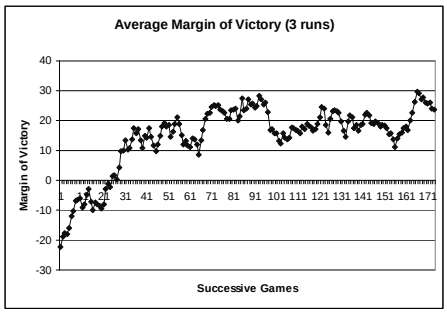

Figure 7: Agent performance learning from past successes and failures

Two learned tactics had a significant impact on performance. First, the agent learned to predict enemy actions and thus dodge short-range enemy missile attacks before they occurred. Second, the agent learned to back away from an enemy while firing its missiles. This opened up future opportunities to dodge (moved to the side) if the agent was currently blocked on both sides.

Conclusion

This work defines the design space for episodic memory and provides an implementation of episodic memory within a cognitive architecture. We have also demonstrated that episodic memory is a useful architectural component and can support many important cognitive capabilities. Each of these capabilities could be implemented individually using various techniques but, inevitably, these implementations must include redundant functionality that is captured singularly in episodic memory.

Episodic memory and similar case-based learning methods are "lazy" while methods such as reinforcement learning are "eager." Lazy mechanisms do minimal generalization so nothing is lost. Generalization and combination with new knowledge is possible at retrieval, but at some computational expense. Eager methods generalize immediately, trading some of the knowledge for efficient application. Thus, lazy methods provide maximal flexibility and are useful when the agent does not know how it will use what it has learned. Eager methods are best when the agent knows how it will use the knowledge. Both approaches are important and an agent should have a combination of these approaches available.

There are many challenges ahead. Our plans include:

1. Demonstrate the usefulness of episodic memory for additional cognitive capabilities. An important part of these demonstrations is developing general procedural knowledge so that a given capability can be used across many tasks without starting from scratch. The implementations of the current capabilities approach this, but we need to test their generality across a range of tasks. For us, the most interesting capability is the interaction of episodic memory with other learning mechanisms.
2. Explore algorithms and data structures for efficient implementations of episodic memory, including hardware solutions.
3. Explore the design space of episodic memory systems. The current design has been successful, but we have not investigated dynamic storage (forgetting and generalization), spontaneous retrieval, and meta-data.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 0413013.

References

Anderson, J. R. & Lebiere C. (1998) *The Atomic Components of Thought*. Mahwah, NJ: Lawrence Erlbaum Associates.

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, Volume 2792, 182-191.

Kolodner, J. (1993) *Case-Based Reasoning*, Morgan Kaufmann Publishers, San Mateo, CA.

Newell, A. (1990) *Unified Theories of Cognition*. Harvard University Press, Cambridge, Mass.

Nuxoll, A. & Laird, J. (2004). A Cognitive Model of Episodic Memory Integrated With a General Cognitive Architecture. International Conference on Cognitive Modeling 2004.

Nuxoll, A., Laird, J., James, M. (2004). Comprehensive Working Memory Activation in Soar. International Conference on Cognitive Modeling.

Ram, A. and Santamaría, J.C. (1997) Continuous Case-Based Reasoning. Artificial Intelligence, (90)1-2:25--77.

Tulving, E. (1983). *Elements of Episodic Memory*. Oxford: Clarendon Press.

Vere, S. and Bickmore, T. (1990). A Basic Agent. *Computational Intelligence*. 6, 41-60.