# A Cognitive Model of Episodic Memory Integrated With a General Cognitive Architecture

**Andrew Nuxoll (anuxoll@umich.edu)**
University of Michigan, 1101 Beal Ave.
Ann Arbor MI 48109-2110 USA

**John E. Laird (laird@umich.edu)**
University of Michigan, 1101 Beal Ave.
Ann Arbor MI 48109-2110 USA

**Abstract**

Episodic memory provides a mechanism for accessing past experiences and has been relatively ignored in computational models of cognition. In this paper, we present a framework for describing the functional stages for computational models of episodic memory: encoding, storage, retrieval and use of the retrieved memories. We present two implementations of a computational model of episodic memory in Soar. We demonstrate all four stages of the model for a simple interactive task.

## Introduction

Episodic memory, as described by Tulving (1983, 2002), captures a history of what happens to an entity. In contrast to semantic learning, episodic learning remembers events and history that are embedded in experience, while semantic learning attempts to extract out facts from their experiential context. Thus a memory of looking out over the grand canyon during your last family vacation is an episodic memory; however, if someone asks what state the Grand Canyon is in, we would typically use semantic memory to answer (unless the answer relies on the recall of a specific episode where someone is telling you where the Grand Canyon is).

In humans, episodic memory has several characteristics that define it and may have an impact on its implementation. Below are some of the most distinguishing aspects of episodic learning:

1. Architectural: The mechanism is used for all tasks and does not compete with knowledge-based reasoning. Reasoning can impact episodic memory by affecting the determination of what is stored such as through deliberate rehearsal.
2. Automatic: Memories are created without a deliberate decision.
3. Autonoetic: A retrieved memory is distinguished from current sensing.
4. Autobiographical: The rememberer remembers the episode from his or her own perspective.
5. Variable Duration: The time period spanned by a memory is not fixed.
6. Temporally Indexed: The rememberer has a sense of the time when the episode occurred.

The obvious value of episodic learning is that it gives an entity a personal history. The recorded history can later be recalled to answer questions about the past, to aid in decision making through predicting the outcome of possible courses of action, possibly by creating an internal model of the entity and its environment, or to help keep track of progress on long-term goals. The history can also be used for deliberate reflection about past events that can improve behavior through other types of learning, such as reinforcement learning (such as the classic “I can’t believe I was so stupid! I’ll never do that again.”). The movie *Memento* (Nolan 2001) gives a taste for how crippled humans would be without it.

There are many challenges in creating computational episodic memory, but the one that stands out is the design of the episodic memory storage and retrieval system. By its very nature, the possible cues for retrieving an episode are not known when it is stored – will your memory of dancing at your high school prom be useful for remembering the name of your date, your favorite song that was playing in the background, or how uncomfortable you felt in that pair of shoes? A further challenge is that, the number of memories continually increases as new episodes are stored without significantly degrading the time it takes to retrieve specific memories.

The goal of our research is to explore models of episodic learning that are useful both for creating AI entities and for modeling human behavior. From the AI side, current agents have no general automatic mechanisms for remembering and accessing their past. From the cognitive modeling side, current models are restricted to single tasks where prior behavior is either forgotten or must be deliberately remembered. Although in the future it may be worthwhile to deviate from some of the details of human episodic learning when building artificial systems, there is such a dearth of understanding of both human and artificial episodic learning, it is worthwhile to use whatever sources of knowledge and research available to advance our understanding of both.

## Prior Work

Episodic memory was first described by Endel Tulving in the early 1970s and later formalized in his book *Elements of Episodic Memory* (Tulving 1983). Research on episodic memory has concentrated either on the phenomenological aspects of the behavior or studies to find a neurological basis for it (Baddeley et al. 2002). Unfortunately there have not been any comprehensive computational models of episodic memory in the psychology literature and AI has paid scant attention to episodic learning. The closest work in AI has been research on case-based reasoning (Kolodner 1993). Episodic memory can be thought of as the mother-of-all case-based reasoning problems – how to store and retrieve cases about everything relevant in an entity's existence. Most case-based reasoning research has avoided these issues. One exception is a Continuous Case-Based Reasoning agent created for a robot navigation domain (Ram & Santamaría 1997). The robot recorded its sensory input at each time increment along with the current distance from its goal. By comparing a series of previous actions and their results to the most recent actions it had taken, it was able to improve its navigation. Two key differences are that their agent relies upon the cases being represented as purely quantitive data and that it performed a match for retrieval using multiple temporally contiguous cases (episodes). To date their work has not been applied outside of the original navigation domain.

There are few instances in which AI systems have an episodic memory (Vere and Bickmore 1990; Rhodes 1997). Vere Basic Agent maintained a limited episodic memory of events. However, the memory did not address issues of how to search and retrieve episodes effectively and efficiently. Rhodes created a wearable computer that continually recorded incoming information and provided the ability to search through the recorded history, but the memory was not integrated in a reasoning architecture and had a preset class of cues that could be used for retrieval. In the cognitive science community, Altmann & John (1999) built an episodic memory model using the Soar architecture that was based upon the recorded behavior of a computer programmer. However, the model was built specifically for that task and did not include a general architectural episodic learning mechanism (it was neither architectural nor automatic and required deliberate processing that could compete with the task at hand). ACT-R's (Anderson & Lebiere, 1998) mechanism for creating new chunks has some characteristics of episodic memory (it stores away partial contents of working memory for subsequent retrieval), but it does not store complete descriptions of working memory, nor are there any methods for retrieving temporally related items, and once a memory is retrieved it is not distinguished as a memory of a prior event.

## Methodology

Our research can be viewed as a search through the design space of episodic learning systems, where different choices for the components (encoding, storage, retrieval) lead to different systems with different capabilities and properties. Our plan is to explore the space, using constraints from what is generally known about human episodic memory, with additional constraints coming from the need to create a computational model embedded in a general cognitive architecture. We have adopted an iterative design methodology, developing models that address some, but not all of the constraints, and then refining them. For example, our models do not encode memories that are temporally indexed or have variable duration.

Our models of episodic learning are *architectural*, meaning that episodic learning will be part of the underlying cognitive architecture and thus available for all tasks. In addition, an architectural learning mechanism does not interfere with other reasoning tasks. Although the underlying implementation is architectural and task independent, what is learned and retrieved will be influenced by task knowledge – both the situation when memories are created and stored, and the situation when memories are retrieved.

Our models of episodic memory are integrated in the Soar cognitive architecture (Newell 1990). Soar is a general cognitive architecture that has been used to model a wide variety of phenomena. It shares many of the features of other architectures such as ACT-R and EPIC (Kieras & Meyer 1997), having a short-term declarative memory and a long-term procedural knowledge encoded as rules. The version of Soar we are using has been extended to include activation of working memory elements (Nuxoll, et. al 2004),

## Framework for Episodic Memory

Episodic learning can be decomposed into the following major phases: encoding – how a memory is captured and stored; storage – how a memory is maintained; retrieval – how a memory is retrieved; and use – how the memory is used to improve behavior. Some of the most basic design decisions, such as the structure of a memory, have impact across all phases.

**Encoding**

- Encoding initiation: when an episode is recorded. Initiation must be automatic, but there are many possible events that could trigger encoding.
- Episode determination: what information is stored in an episode.
- Feature selection: what features in an episode will be available for retrieval.

**Storage**

- Episode structure: how the encoded episode is stored.
- Episode dynamics: how stored episodes change over time (such as possible decay and removal).

**Retrieval**

- Retrieval initiation: how retrieval is triggered – automatically and/or deliberately.
- Cue determination: which data are used to cue the retrieval of an episode.
- Retrieval: which episode is retrieved given the current cue.
- Retrieved episode representation: how the retrieved episode is represented.
- Retrieval meta-data: what meta-information about the retrieved episode is available.

**Use**

- Once the episode is retrieved, how it is used to aid reasoning. This is not a part of the design of the episodic learning system, but depends on the capabilities of the embedding architecture, general methods, and task knowledge.

## Pilot Implementation

Our first implementation was an attempt to confront and explore the issues that arise when developing a comprehensive, architectural episodic learning mechanism, without worrying about getting all the details right. To speed the implementation, we did what was easiest to do within Soar. Thus, episodes were based on the contents of working memory and episodes were stored as rules, with a subset of the current situation being designated as features used for retrieval, while other aspects of the situation were designated as the content of the retrieved episode.

Originally, the conditions and actions of the rules were based on a hand-selected subset of working memory that we knew was optimal for our domain. We experimented with using only the most highly activated working memory elements, but then faced the expected problem that if too few features were selected, the retrieval was too general in situations that were not appropriate, whereas if too many features were selected, the retrieval was overly specific and did not provide significant transfer to new situations. We were unable to develop a task-independent method to make sure that just the right number was recorded.

Although the use of rules for memories was limiting, we completed an end-to-end episodic memory system. Memories were automatically recorded whenever there was a significant change in the highest activated working memory elements. Memories were retrieved when a special symbol was created in working memory to trigger retrieval (all episodic memory rules tested for the existence of this symbol). The retrieved memories were created in an agent-specified area of working memory so that they could be distinguished from the rest of working memory.

There could be many uses of the episodes, but in the agents we developed, the episodes include sufficient information for the agent to determine what feedback it would get from the environment if it did the same action in the same situation in the future. The agents can use the episodes in one of two ways. One way aids decision making by using retrieved episodes to predict the results of the actions and then choosing the action with the best predicted result. In the other case, episodes were retrieved as the entity deliberately reflected about the task, generating possible situations and recalling the results of its actions. In both cases, Soar's chunking mechanism learned control rules to speed future decision making.

The strength of this implementation is that it demonstrated that it is possible to create an architectural mechanism that automatically creates episodic memories and that those memories can be recalled in the future to improve problem solving without confusing the agent (partially fulfilling the first four properties of episodic memories). Its major failing arose from encoding the episodes as rules that required complete matches in order to retrieve the memories, which forced us to handpick the features encoded in the memory.

## Current Implementation

The most significant change in the current implementation is that we have created a separate episodic memory in Soar, one where complete episodes are stored (similar to cases in case-based reasoning). Retrieval of the episodes is patterned on the use of buffers in ACT-R, where a retrieval is made when a cue is placed in a special location in working memory. The retrieval is based on a partial matching algorithm. This design removes all domain-dependent aspects of the previous system. The new episodic memory system fits within our episodic memory framework as follows:

**Encoding:**

- Encoding initiation. A new memory is encoded each time the agent takes an action in the world.
- Episode determination. All working memory elements, including their activations, are included in the episode.
- Feature selection. All features in the episode can participate in retrieval, so no feature selection is made.

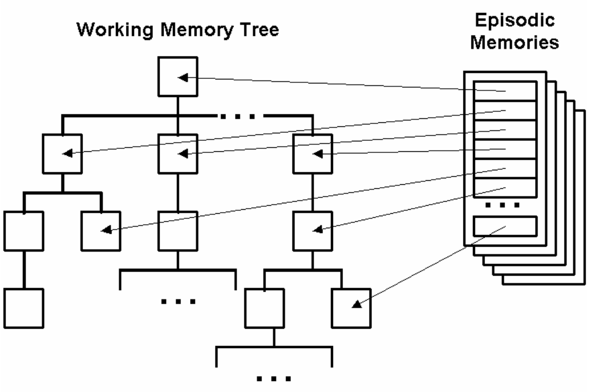


Figure 1: Episodic Memory Structure

**Storage**

- Episode structure. As shown in Figure 1, our episodic memory structure attempts to minimize the overall storage requirements by building up a single structure (the working memory tree) that holds a single instance of all elements that have ever been in working memory. Each episode consists of a list of pointers to each of the working memory elements it contains, in a canonical order. In addition, all elements in the tree have pointers to the memories that contain them. (This is not depicted in the diagram.) The activation level of each element at the time of storage is also recorded.

- Episode dynamics. Currently, episodic memories do not change over time.

**Retrieval**

- Retrieval initiation. Retrieval is initiated deliberately by placing a cue in a special location in working memory.
- Cue determination. A cue is deliberately constructed by the rest of the cognitive system (using rules) in a reserved location in working memory. The cue can have any number of working memory elements.
- Retrieval. The cue is compared to all stored episodes, selecting the episode that "best matches" the cue. The match is determined by totaling the number of working memory elements that are shared between the cue and the episode. The match uses the following algorithm:
  1. The system traverses the tree containing all known working memory elements (see **Figure 1**) finding all of the elements in the cue.
  2. Each element that is matched in the tree contains a list of references to every episode that contains it. (These references are not depicted in the figure.) A list of all episodic memories that contain at least one element from the cue is created. by merging the references from each matched part of the cue.
  3. The complete cue is then compared to each episodic memory in the episodic memory list and the one that best matches the cue is selected.

  Because the number of episodic memories in the system continues to grow and any memory may potentially match a given cue, the retrieval phase may slow as more episodes are stored. However, this above approach was straightforward to implement and allowed us to create a working episodic memory system rather quickly.

  Once an episode has been retrieved, the system can also retrieve a series of episodes in temporal order via a special "next episode" command.
- Retrieved episode representation. The complete episode is retrieved in a labeled area of working memory to avoid confusion with the current state of the agent.
- Retrieval meta-data. Currently there is no meta-data retrieved with the episode.

**Use**

- Initially our implementation is used to support the same use as in the pilot implementation: evaluation of alternative actions. The following section describes this in detail for a simple interactive task.

## The Eaters Environment

To test the system, we created episodic memory agents with different episodic memory variants in a simple game called Eaters. An eater is a Pacman-like agent that moves around a 16x16 grid world. Each cell in the grid is either empty or contains a wall, normal food (● = 5pts), or bonus food (■ = 10 pts). 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 eats the food (receiving the appropriate score). When an eater leaves a cell it becomes an empty cell. 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, its current score, its color, and number of moves taken so far. The figure below depicts the input available to an eater (although it is represented completely symbolically for the eater).

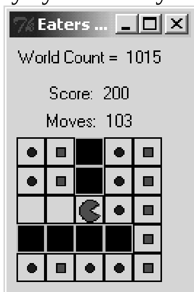


Figure 2: An eater's sensory input

To test the episodic memory system, we created an eater to run in Soar-EM (Soar version 8.5 extended to include the episodic memory system). This eater does not know the relative value of the objects it senses, nor does it have a model of how its actions move it through the world. Our goal is for it to use its episodic memory in place of that knowledge to aid in selecting which direction it should move. For each of the possible directions, it creates a memory cue composed of its current sensory input and the proposed direction of travel. Once it has retrieved a memory of prior situations, it then retrieves the next memory (in temporal order). This new memory includes the score it received after taking the proposed action in the retrieved situation. The change in score between the two memories provides a quantitative evaluation for the proposed action. This evaluation is used to compare the proposed action to the other possible actions (whose evaluations are computed in the same manner). The agent selects the action with the highest evaluation.

There is no guarantee that an appropriate prior episode will be retrieved – that will depend on the set of recorded episodes and the algorithm used for retrieving episodes from memory. If no prior memory is found the agent must still assign a default evaluation to the action so that it can compare the action to the other possible actions. If the default value is greater than bonus food, the eater will be biased to explore new actions, which in turn would build up its episodic memory. If a low value is used, the eater will avoid the unknown. Although we tested the eaters with a variety of values, the differences in behavior were slight and the results we report are for the eater that favored exploration.

## Results

To evaluate the effectiveness of episodic memory, we created two baseline eaters (without episodic memory) for comparison: a random eater and a greedy eater. The random eater has no knowledge of the value of cells and selects its next move randomly. The greedy eater always selects the move that will bring it the highest immediate score. The random eater serves as a lower bound on performance while the greedy eater provides an approximation to an upper bound on performance.

Our first episodic memory system implementation ranked the episodes by the number of features matched against the cue, with the episodes having the highest number of matching features picked. Ties were broken randomly. In this task, there are a total of 31-36 features per state; however, the correct decision is determined by the specific values of only two features: the direction being moved and the content of the cell being moved into. To explore the impact of having the episodic memory filled with different numbers of episodes, the experiment had five iterations where we measure the score of the eater for each iteration. For each iteration, the eater made 1500 internal decisions, which resulted in an average of 130 actions in the world. (Multiple internal decisions are required to evaluate the alternative actions and finally select the external action.) After each iteration, the contents of eater's world map were reset (randomly rearranged and refilled with food) and the score was reset to 0. The eaters were allowed to retain their episodic memories between iterations. The experiment was repeated five times and the results were averaged together.

The first set of bars in **Figure 3** show the episodic memory eaters' average scores after their $100^{th}$ move in each iteration. The dashed lines show the average scores for the greedy and random eaters.

These results suggest that the unbiased eaters (darker columns on left of each pair) benefit from using their episodic memory to select actions. However, this benefit is small as its score is only moderately better than it would receive for random actions. Also, the eaters' performance did not improve significantly on subsequent iterations despite having more memories available.

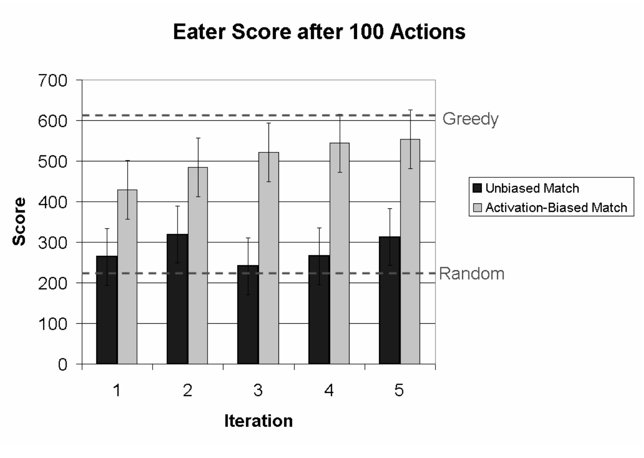


Figure 3: Average score after 100 actions

Another intriguing question is why the eater requires relatively few memories to achieve better than random performance but fails to improve beyond this level. This turns out to be of the result of an interaction between features of the task and our specific implementation. Recall that the eater creates a cue to retrieve a memory for each direction that it is considering. Since these cues differ by only a single feature (the proposed direction of travel), it is often the case that this difference is overwhelmed by a good match with other features of the cue. As a result, the eater recalls the same episode for each direction. Thus, the eater will often behave randomly, particularly when it has fewer memories in its episodic store. The eater is able to perform better than random as it gains more memories because it eventually has a large enough memory store that the single difference between the cue affects the match.

Given that memories are never lost, the expectation that an eater using an unbiased match will eventually achieve greedy behavior is probably true. However, by rough estimation, there are over 1 quadrillion different possible inputs in even this simple domain. Since only two of the features determine the correct match, the agent would have to have a *lot* of memories before its unbiased best match became a reliable predictor.

Clearly, the key problem with an unbiased partial match is that the likelihood of retrieving a truly relevant episode is small because of the large number of irrelevant features that have equal say in whether an episode is retrieved. What is needed is some mechanism that biases the selection toward relevant features. However, the relative value of matching different working memory elements is highly task dependent. A general purpose episodic memory must have a way of ranking features in a domain independent manner. To resolve this issue, we hypothesized that working memory elements which were highly active when the episode was created would be more likely to be relevant.

The activation mechanism used to bias the match is a recent addition to Soar (Nuxoll, Laird & James, 2004) with the following properties.

- Working memory elements receive an initial activation based upon the activation level of existing elements which were tested by the production that created the new element.
- An element receives an activation boost whenever a production fires that has tested it.
- Activation levels decay over time using a formula similar to that used by ACT-R (Anderson and Lebiere 1998).

For our second trial, we modified the retrieval mechanism to be biased in favor of episodes that had highly activated features which matched the cue. For each match, the activation values of all matching elements were summed. These sums were compared and the highest value was selected as the best match.

The previous experiment was repeated with the new activation-based matching scheme. The results are shown as the second set of bars in **Figure 3**. The episodic memory eater with an activation-based match begins by performing significantly better than its unbiased match predecessor. Furthermore, as it gains more memories the eater's performance continues to improve until it performs comparably to the greedy eater.

An alternative view of the results is in **Figure 4**. This graph depicts the fraction of correct evaluations that the agent has made over the course of an entire run (five iterations of 1500 cycles). As before, this data is an average of five runs. Both agents made over 2200 actions during the run. This graph shows that the eater with activation-biased match is improving its ability to assess its situation at a consistently faster rate than the eater with unbiased match.

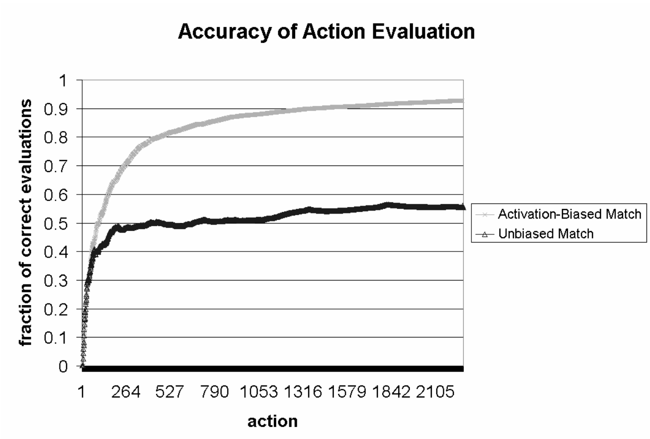


Figure 4 Comparison of eater action evaluation

## Issues and Future Work

We obviously need to apply this model to more complex tasks and start comparing to human data, although it is difficult to find experiments that single out only episodic memory. Beyond that, the episodic memory framework we've established provides a good basis for examining issues and shortcomings in the current model and providing an agenda for future research.

**Encoding**

- Encoding initiation: As an implementation convenience we abandoned the pilot approach of using significant changes in activation to trigger an encoding. It is unclear what is the "right" trigger for initiation and we need to explore our original approach as well as others.
- Episode determination: We currently store and retrieve the complete state of working memory. As part of our attempts to improve efficiency, we may need to store and/or retrieve only subsets of working memory.

**Storage**

- Episode structure: We may need to modify the structure to improve retrieval efficiency. We also need to find ways of encoding temporal information in the episodes, which we expect to be more difficult than just recording the time of storage. The temporal qualities of retrieved episodes appear to be more subtle than time tags.
- Episode dynamics: Human memory is rarely eidetic. How is it that some parts of memories are lost and others retained?

**Retrieval**

- Retrieval initiation: Initiation is always deliberate based on a specific cue. Another possibility is automatic cuing based on the contents of working memory.
- Retrieval: In the worst case retrieval is linear with the number of stored episodes, which will greatly limit the usefulness of episodic memory. We need to develop algorithms that approach constant time, although this may be possible only with a parallel implementation.

**Use**

- There are many avenues for using episodic memory for improving agent behavior. Some of our future efforts will include:
  - creating episodic agents in more environments where the relationships between cues and episodes is much more complex;
  - exploring the relationship between episodic and semantic memory;
  - using of episodic memory in reflective learning;
  - exploring the influence of affect on episodic memory.

## References


Altmann, E. M. & John, B.E. (1999) Episodic indexing: A model of memory for attention events. Cognitive Science, 23, 117-156.

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

Baddeley, A., Conway, M. & Aggleton, J. (2002) Episodic Memory: New directions in research. Oxford University Press, Oxford.

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.

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

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

Nolan, C. (Director) (2000) Memento [Film]. Newmarket Films. http://us.imdb.com/Title?0209144

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.

Rhodes, B.J. (1997) The Wearable Remembrance Agent: A System for Augmented Memory. Personal Technologies.

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

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

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