Advances in Cognitive Systems 4 (2016) 1-12 Submitted 4/2016; published 6/2016

Architectural Mechanisms for Mitigating Uncertainty during Long-Term Declarative Knowledge Access

Justin Li JUSTINNHLI@OXY.EDU
Cognitive Science, Occidental College, Los Angeles, CA 90041 USA

Steven Jones SCIJONES@UMICH.EDU
Computer Science and Engineering, University of Michigan, Ann Arbor, MI 48109 USA

Shiwali Mohan SHIWALI.MOHAN@PARC.COM
Palo Alto Research Center, Palo Alto, CA 94304 USA

Nate Derbinsky DERBINSKYN@WIT.EDU
Computing Science and Networking, Wentworth Institute of Technology, Boston, MA 02115 USA

Abstract

Long-term declarative semantic memory is a key component of a cognitive architecture that allows an agent to perform effectively in complex tasks. Over the years, a number of new mechanisms beyond deliberate cued retrieval have been developed, including partial matching and spreading activation. However, there has yet to be a theory that coherently explains why these mechanisms are beneficial or necessary. We propose that these semantic memory mechanisms are attempts to mitigate the uncertainty in the environment, as well as to compensate for the errors and incompleteness in agent knowledge. Through this lens, we show that existing mechanisms are misunderstood and motivate future areas of exploration, including how retrieval mechanisms should incorporate agent problem-solving context and how to unify the treatment of diverse mechanisms and the uncertainties they address.

1. Introduction

Semantic memory, as a large store of general knowledge, has become a part of the prototypical cognitive architecture. Originally incorporated into ACT-R, similar architectural components have since been incorporated into Soar, Sigma, and others, and have been used in language, perception, reasoning, learning, and a myriad of other tasks. As a result, a number of new semantic memory mechanisms have been developed over the years, which improve on the algorithms that determine which piece of knowledge to retrieve, and provide agents with new ways to specify the desired knowledge.

One area that has lagged behind, however, is an overarching theory of how these memory mechanisms are functionally beneficial. Each mechanism has been evaluated with particular tasks, but no guiding principle has emerged as to why new memory mechanisms are consistently necessary,

© 2016 Cognitive Systems Foundation. All rights reserved.

nor as to when semantic memory may be considered complete. While the rational analysis of memory defines the rough goals of a memory system, its application in deriving base-level activation (Anderson & Schooler, 1991) only considered the environment, but omitted the agent from the narrative and thus incompletely explains the diversity of mechanisms. We propose that many of these mechanisms are in fact attempts to deal with uncertainties that arise from a combination of the partial observability of the environment and a lack of various forms of agent knowledge. Although the memory system cannot completely eliminate these uncertainties, it could be designed to incorporate more information into deciding which piece of knowledge is relevant to the current situation.

In the rest of this paper, we first recall the motivation for long-term memory and show that the deliberate cued retrieval mechanism in fact introduces uncertainties. Then, the bulk of the paper explores how the diversity of memory mechanisms may be understood as mitigating forms of uncertainties, while identifying the architectural components that are used. By explicitly considering uncertainty as one of the computational constraints that cognitive architectures must address, we provide a new framework for understanding the role of these mechanisms in memory retrieval and identify future areas of exploration for how retrieval mechanisms should incorporate agent problem-solving context.

2. Uncertainty in Long-Term Memory

In this section, we describe the prototypical cognitive architecture, which includes a long-term memory and a deliberate cued retrieval mechanism. This description is biased towards Soar (Laird, 2012), although the broad strokes also apply to other cognitive architectures such as ACT-R (Anderson, Bothell, Byrne, Douglass, Lebiere, & Qin, 2004).

Declarative knowledge is represented as directed, edge-labeled graphs, where individual units of knowledge (i.e. graph edges) are termed *memory elements*. *Working memory* is one such graph that contains knowledge relevant to the agent's immediate situation, including the current goal and any knowledge related to current reasoning. Fixed regions of the graph represent the perceptual input and motor output of the agent (similar to "buffers" in ACT-R), which allows agent reasoning to incorporate and change aspects of the external world. Working memory is modified by *procedural knowledge*, represented as a set of if-then rules that match on elements in working memory. When a rule matches and fires, working memory is modified according to the action side of the rule, which may add and/or remove any number of memory elements.

In addition to graph regions that facilitate interaction with the environment, working memory also contains regions for interaction with other architectural components, including *long-term memory* (LTM). While Soar has two declarative LTMs, Episodic Memory and Semantic Memory, this paper focuses on the latter, which holds many similarities to the Declarative Memory module of ACT-R. Semantic memory is represented as a set of memory elements, containing general facts about the world, independent of the context in which they were originally learned. Since this knowledge base may grow very large, and many of these memory elements may not be relevant to the agent's current situation, semantic knowledge is not directly accessible to procedural knowledge, and thus cannot directly affect agent behavior. Instead, semantic memory elements must first be *retrieved* into working memory. To retrieve semantic knowledge, the agent must first create a *cue*, a node in working memory that describes, via outgoing edges, a subset of the features of the desired node in semantic memory. Given this cue, the architecture then selects one node in semantic memory that "best" matches the cue, and copies the node and its outgoing edges into a fixed region of working memory.

The dissociation of semantic knowledge from working memory is not a historical artifact, but is motivated by two functional constraints on knowledge-rich agents that operate in real time:

1. In order for an agent to complete tasks in dynamic environments, it must be sufficiently reactive to changes in the environment. From human studies and empirical system-building in a variety of domains, a common threshold for reactivity is 50 milliseconds (Rosenbloom, 2012); exceeding that limit when making a decision may lead to suboptimal behavior.

2. Long-lived agents contending with multiple, complex tasks must acquire and make use of large amounts of knowledge. Much of this experience will likely be necessary to behave effectively, but most of which is likely irrelevant for making any individual decision.

Long-term memory is necessary if an agent is to maintain both reactivity and access arbitrarily large knowledge bases. Thus, long-term memory serves as a necessary tradeoff: the agent is able to accumulate large amounts of knowledge, but is limited in how much is directly accessible to reasoning, thereby avoiding computational explosion (Derbinsky & Laird, 2010).

Although this line of reasoning is not new, we reiterate it here for two reasons. First, it serves as an example of architectural development that is guided by computational constraints imposed by the environment and the task. Second, we note that (to our knowledge) no similar justification for the deliberate cued retrieval mechanism exists, aside from inspiration drawn from studies of human semantic memory phenomenon. Rather, the mechanism arose as an "obvious" solution for how to transfer knowledge between the two memories. This is problematic, as it implies that the consequences of such a mechanism may not have been fully taken into account.

In particular, a deliberate cued retrieval mechanism assumes that the agent can recognize that additional knowledge is needed and can correctly identify the characteristics of the requisite knowledge. In practice, these assumptions may not hold in continuous, partially observable environments, where the agent's procedural and/or short-term knowledge may be erroneous or incomplete. This may be expressed as uncertainty on the part of the agent, and for the problem of retrieval knowledge from LTM, these uncertainties manifest in at least three ways:

1. *Uncertainty about how to construct the cue*: The agent's LTM is growing non-monotonically as it learns and reasons about its environment. Agent knowledge to access and reason with LTM may become outdated as a result. Over time, the cue generated by procedural knowledge may become over-specific as memories, or their features, are deleted from LTM, or become under-specific as LTM learns new elements that are common across multiple LTM nodes.

2. *Uncertainty about when to cue the memory*: The agent may not always recognize that it needs additional knowledge from LTM to produce behavior in a situation. This is particularly true if the agent is learning, and the retrieval of newly acquired knowledge would allow higher performance where previously no retrieval was necessary (Gorski, 2012). This may

create situations where the agent continues without accessing memory that could influence its behavior or performance.

3. *Uncertainty about how to represent a situation*: Partially-observable environments, sensory noise, and incomplete reasoning knowledge can all cause the agent to represent a situation incorrectly or incompletely, potentially leading to erroneous retrieval cues for LTM and compounding the inaccuracy.

While there is considerable literature on long-term memory mechanisms, including systems that address a subset of these concerns — most prominently, the possibility of an ambiguous cue — relatively little work has considered how memory should address representational uncertainties. The central proposal of this paper is that LTM mechanisms are best understood as attempts to mitigate these uncertainties and that explicitly considering the uncertainty that a memory mechanism is addressing may provide clarity on how the mechanism should function.

3. Memory Mechanisms for Mitigating Uncertainty

This main section of the paper considers four memory mechanisms and examines what uncertainty they address. In addition to tracing why the mechanism may be useful within this framework, we also consider where the mechanism draws information, which we assemble into a model of agent memory in the conclusion.

3.1 Partial Match and Spontaneous Retrieval

In humans, memory can be directed to search for knowledge about particular objects or with particular properties. In a cognitive architecture, this interaction takes the form of a cued retrieval, where the agent can guide which memory is retrieved through a cue that describes the features of the desired node in long-term memory. The cue acts as a hard constraint on the retrieval process — only nodes that satisfy all described features are considered for retrieval, and the retrieval may result in failure if no node matches all the features.

The cued retrieval mechanism could be hampered by all three types of uncertainty listed in Section 2. First, the agent may not have sufficient information to create a cue that uniquely identifies a piece of knowledge, resulting in an under-specified cue that multiple candidate memories may all satisfy. The memory-retrieval mechanism must therefore use additional information, aside from the cue, to bias which node is retrieved. What information exists and how it is used is addressed in the next two sections.

The opposite problem could also occur if the cue was created based on incorrect knowledge, or through a misrepresentation of the situation. In both of these cases, the cue may exclude useful knowledge — or in the extreme case, it may exclude from retrieval *all* knowledge in long-term memory. This type of error is addressed by the *partial-match* mechanism found in the declarative memory of ACT-R and in the episodic memory of Soar. In partial match, the hard constraint that the retrieved node must contain all cue features is relaxed; however, the exact relaxation differs between architectures. In ACT-R, candidate memories are instead evaluated by a *partial match score*, which represents the similarity (or "difference penalty") between a retrieval candidate and the cue (Bothell, 2013). The more similar a memory is to the cue, the more likely it will be retrieved, allowing the architecture to model arithmetic errors and confusion between semantically related concepts.

A different model of partial match may be found in Soar's episodic memory, where memories satisfying *any* cue feature are considered, instead of only memories that satisfy *all* cue features (Derbinsky & Laird, 2009). As with partial match in ACT-R, partial match in Soar's episodic memory also affects retrieval; in this case, memories that match more of the cue are more likely to be retrieved.

It is instructive to consider how the mechanisms of ACT-R and Soar address different types of error in the cue. The similarity-based partial matching of ACT-R reflects the possibility of an incorrect cue, such as using a wrong number in an arithmetic fact, or substituting semantically related concepts for each other. The modeler-defined similarity function allows ACT-R's declarative memory to deduce what is most likely to be the intended desired memory. By contrast, Soar's partial-matching mechanism attempts to account for the agent over-specifying the cue to describe useful features that may never have co-occurred at any time. This mechanism assumes a domain-independent similarity function on the graph structure of knowledge, unlike the domain-dependent similarity function in ACT-R. Regardless, in both cases, partial matching ameliorates one class of agent error, with different architectures currently supporting different types of errors.

The retrieval mechanism thus far assumes that the agent is able to deliberately create a cue for retrieval, even if the agent lacks the knowledge to create a cue that identifies exactly one memory. However, the agent may be uncertain about when is the appropriate situation to deliberately begin a cued retrieval. Long-term memory can mitigate this situation by automatically, or *spontaneously*, retrieving potentially relevant knowledge into working memory. This is equivalent to the *buffer stuffing* mechanism for ACT-R's perceptual buffers (Bothell, 2013) and the spontaneous retrieval mechanism in Soar (Li & Laird, 2015). Both mechanisms have only been developed recently, which provides evidence for our hypothesis that changes in memory mechanisms are partially in response to ameliorating various uncertainties.

All three mechanisms described in this section — standard deliberate cued retrieval, partial matching, and spontaneous uncued retrieval — do not exclude the possibility that multiple memories may equally match the cue (if the cue exists at all). We emphasize that this is not necessarily due to any flaw in agent design, but may be a result of environmental complexity and/or partial observability. Regardless, if the cue (as the most concrete signal of retrieval) cannot uniquely identify the node to be retrieved, additional memory mechanisms must be in place to disambiguate which node is desired.

3.2 Base-Level Activation

The most common source of information used by the architecture to select which of multiple matching memories to retrieve is the retrieval history of a memory; implicit is the assumption that what the agent retrieved in the past is likely to be what the agent is currently trying to retrieve. In Soar, ACT-R, and other architectures, the retrieval history is often encapsulated in a single *activation* value and although the exact meaning of activation depends on architectural settings, the use of *base-level activation* is standard. The base-level activation of a memory increases with the frequency and recency of use, while decaying over time, and has been well established as useful in domains such as linguistics (Anderson & Schooler, 1991; Derbinsky & Laird, 2011).

In the context of long-term memory compensating for agent uncertainty, we note that retrieval history itself is ambiguous in meaning: that a node has been retrieved before does not mean that the node's knowledge was desired — in fact, it is not uncommon for agents to retrieve multiple irrelevant memory nodes before finding a useful one. While this pattern suggests the possibility of new mechanisms for agents to indicate the desirability of a retrieved memory, it does not fall under the umbrella of agent uncertainty; even without such a mechanism, an agent with knowledge of the operations of its long-term memory could repeatedly retrieve the correct node, solely to boost its activation. Whether memory is reinforced through this kind of *rehearsal* or through another mechanism, more pertinent to this paper is the possibility that the agent may be mistaken in whether a memory is needed. Thus, although base-level activation has a long history of being used to bias memory retrieval, it cannot completely ameliorate the effects of agent uncertainty.

There are additional reasons to incorporate more information into the calculation of retrieval bias. For one, there is no consensus on what should constitute the "use" of a memory node. Currently, both Soar and ACT-R boost the activation of a node whenever it and its elements are stored into long-term memory (deliberately or automatically) and whenever it is the result of a retrieval. It may also be possible to take into account the length of time knowledge is in working memory; ACT-R's automatic storage mechanism considers whether a node has been in working memory, but fails to consider its duration of stay. For Soar, a separate *working-memory activation* exists, which boosts activation based on whether an element is used in the condition of a firing rule (Nuxoll, Laird, & James, 2004; Derbinsky & Laird, 2012). Both mechanisms suggest that the contents of working memory should also be considered during retrieval, which we discuss next.

3.3 Spreading Activation

Spreading activation describes how the activation of a node should depend on the activity of its neighbors. More generally, we say that spreading activation is taking advantage of the *context* of the retrieval, meaning not only the current goal of the agent and other relevant knowledge in working memory, but potentially the retrieval history of *other* long-term memories as well. The intuition behind taking this context into account is that different tasks requiring different knowledge may nonetheless result in exactly the same cue and activation of memories. This may be the case in a word-sense disambiguation task, where neither the cue (a polysemous word) nor the activation (the frequency/recency of word senses) provide information on a specific use of the word. Instead, the correct word sense is determined by the senses of previous words in the sentence — the context in which the disambiguation occurs.

This general idea contributes to the retrieval bias in slightly different ways in ACT-R and Soar. In ACT-R, activation spreads from nodes in working memory at the time of the retrieval, such that nodes in long-term memory that are connected to buffer nodes are more likely to be retrieved. The effect of spreading activation in ACT-R is separate from base-level activation, and has no effect beyond the single decision cycle in which the *current* retrieval occurs (Bothell, 2013).

By contrast, spreading activation in Soar occurs when a long-term memory is stored or retrieved and its base-level activation is boosted (Li & Laird, 2015). At that time, neighboring memories also receive a boost to their base-level activation, so that those nodes are more likely to be retrieved in *future* retrievals, at which point base-level activation is the only bias (assuming no partial matching occurs). Thus, spreading activation in Soar has a temporal effect that spreading in ACT-R does not.

Given that both architectures have spreading-activation mechanisms, it is surprising that there has yet to be a coherent account of spreading activation. This is apparent even in the differences between ACT-R and Soar, in the sources of activation (is it from working memory or long-term memory?) and the long-term effects (does it modify base-level activation?). We believe these fundamental differences reflect a general lack of consensus of what spreading activation represents, and here we propose understanding the mechanism from the point of view of mitigating uncertainty.

We start by considering the common interpretation that spreading activation represents the co-occurrence of memories, and that the presence of one node in working memory is evidence that other correlated nodes are needed, and therefore retrieval should be biased towards those elements. In other words, similar to how base-level activation is related to the prior probability that a memory element is desired, the context of a retrieval is Bayesian evidence for the same. We will show, however, that there are theoretical issues with this explanation, and that it is not reflective of how spreading activation is implemented in ACT-R and Soar.

3.3.1 Spreading as Co-Occurrence

The interpretation of spreading activation as representing co-occurrence is perhaps the natural extension to the proposal of base-level activation as representing the need odds of an item. Since need odds are derived from the probability that an item will occur, spreading may be interpreted "as a reflection of the Bayesian prediction of the likelihood of one [node] in the context of other [nodes]" (Pirolli, 2005). Since the existence of other nodes in working memory is unambiguous, spreading activation may provide additional information as to which memory is useful; this may also be used to model priming effects in people.

While the introduction of Bayesian reasoning (through the likelihood) is welcome, we note that this alone does not resolve the differences between how spreading activation is used in ACT-R and Soar. The Bayesian framework highlights the assumption that each architecture has made. In Soar, the temporal effects of spreading activation suggest that the correlation between memories that are being considered is not just co-occurrence, but also the sequential presentation of the nodes (even if they do not co-occur). This is why the previous contents of working memory (that is, memories that were retrieved from a previous decision cycle) could affect the result of a future retrieval. In contrast, ACT-R does not consider such relationships as contributing to whether a memory is useful (although there has been work towards incorporating this information (Thomson & Lebiere, 2013)), and thus correlation is restricted to the actual co-occurrence of nodes in working memory. These differences are not necessarily in the Bayesian interpretation of spreading activation, but in what constitutes the context of a retrieval, whether it is only the nodes that are currently elements in working memory, or if the context also includes the recent history of working memory. Reconciling this difference is beyond the scope of this paper.

Regardless of what is considered the full context of a retrieval, spreading activation in neither architecture fully commits to the Bayesian interpretation. We make this claim based on two major ways in which the implementation of spreading activation fails to conform to what a Bayesian analysis would require. The first divergence is that at no point in either architecture is the correlation between arbitrary sets of memories truly taken into account. If spreading activation were to represent the likelihood of one node given any context, the architecture would have to keep track of the full joint distribution of all nodes in long-term memory. This is prohibitive in both computation and memory, but neither architecture attempts any factorization of this distribution to a Bayesian network, nor do the architectures store pairwise correlation information. At best, this suggests that the current implementations of spreading activation is some approximation of the Bayesian likelihood; at worst, a Bayesian memory is only a stated goal, but is not meaningful in practice.

The second divergence from Bayesian theory builds on the first, but approached from the viewpoint of the implementation. In both ACT-R and Soar, spreading activation occurs on the edges of the semantic network, which explicitly do not represent causality. In Soar, these edges may come from knowledge bases (KBs) that have been loaded into semantic memory, which could represent taxonomic (from Cyc or DBpedia) or lexical (from WordNet) information (Lenat, 1995; Bizer, Lehmann, Kobilarov, Auer, Becker, Cyganiak, & Hellmann, 2009; Miller, 1995). If no KBs are used, the edges are likely to be the result of how the agent designer chose to represent information in the agent, with no consideration as to whether the nodes are causally related. If spreading activation truly reflects correlational or causal relationships, there is no reason activation should spread along a semantic network.

Although we agree with existing literature that correlation should be taken into account in long-term memory retrievals, the existing implementation of ACT-R and Soar suggests that spreading activation is, in fact, representing a different type of information.

3.3.2 Spreading as Representational Uncertainty

What, then, does spreading activation represent in ACT-R and Soar?

We propose that spreading activation represents *the agent's uncertainty* about whether it is using the correct concepts to represent its experience. This parallels the Bayesian interpretation of kernel density estimation, where points are sampled from a probability space, and the goal is to approximate the probability density function. To generalize from the sampled points, a kernel is applied such that nearby points in similarity space also receive probability weight; the density at one point is therefore the sum of all such weights from the sampled points. This compensates for uncertainty about the samples collected, and remains silent about correlations between neighboring points (although it is not ruled out). The relationship of kernel density estimation to spreading activation is most obvious if we discretize the sample space into a lattice. For any unimodal kernel (which includes the common Gaussian kernel), the weight given to a point would decrease as a function of the graph distance of that point from the sample — much like how the boost from spreading activation is often larger for nearby graph neighbors.

By this analogy, spreading activation in semantic networks plays the same role as kernel functions for density estimation, by providing weight to nearby concepts to allow for errors in the agent's representation. Consider the example of a robin (the bird) — an agent may need to reason about the fact that robins have spinal cords. While this is true, the knowledge used is more generally about vertebrates, thus other facts about vertebrates should receive more activation to indicate that there is a higher probability they are needed. This is a purely semantic relation between the two concepts, one unlikely to come about due to co-occurrence correlations, and therefore unlikely to be learned through statistics.

The utility of boosting the activation of semantically-related concepts is not restricted to ancestor-descendant concepts within an ontology. Other relationships where this may occur include ontological siblings (e.g. misrepresenting cheetahs as jaguars), linguistically substituting an object for its part (e.g. saying that "the laptop died" when it is the battery that is dead), or even substituting physical objects for abstract ones (e.g. "the battery died" is really a statement about energy). These linguistics examples suffice to demonstrate the utility of casting spreading activation as mitigating representational uncertainty, and we note that the problem exists in any domain where percepts and agent reasoning may not be perfectly precise.

Thus, although spreading activation does not directly correspond to Bayesian likelihood, it does embody a Bayesian approach in tackling the uncertainty in memory retrieval. Separating these different interpretations of spreading makes obvious a more general way of taking the retrieval context into account, by also including correlational information together with the representational uncertainty. A complete spreading activation mechanism would look at whether a node is correlated with other nodes, then *weight the result by the probability that the other node is the correct representation of the situation*. In our view, this would fully account for the uncertainty about, and the information inherent to, the context of a retrieval.

4. Model of Memory Metadata

Throughout our examination of how the memory mechanisms mitigate uncertainty, we have noted that there are four main sources of information from which these mechanisms draw:

1. the elements of working memory, used by correlational spreading activation;
2. the retrieval cue created by the agent, used by partial match as a relaxation of the hard constraint;
3. the retrieval history of elements in long-term memory, used by correlational spreading activation and also captured by base-level activation; and
4. the structure of long-term memory, used by structural spreading activation.

Although most of these components of the total available information are already leveraged by existing memory-retrieval implementations to some extent, we are not aware of a framework that unifies the role that each component plays. We do not suggest that retrieval should *only* take these components into account; other parts of the agent state may provide additional information, including metadata associated with procedural rules, the contents of other memory systems (e.g. short-term visual memory, episodic memory), and emotion/appraisal influences. These components have relatively little standardization across architectures, however, so we have only considered the list above.

This list of architectural components that currently contribute to reducing retrieval uncertainty paves the way for a more rigorous analysis of the demands of long-term memory. If we assume some

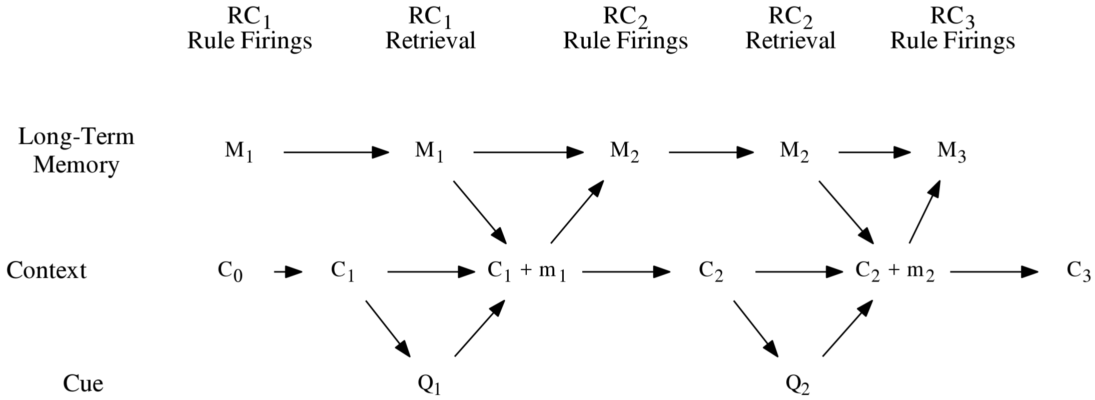

Figure 1: A model of memory processing, with arrows representing influence. M is the information in long-term memory, including both structure and metadata; C is the context of a retrieval (e.g. the contents of working memory); Q is the cue; and m is the retrieved memory. An RC is a retrieval cycle, and each component is numbered with the retrieval cycle to which it belongs.

degree of independence between the components — which is consistent with how these components interact within existing architectures — we can depict the reasoning, learning, and retrieval processes with Figure 1. (The storage of new knowledge is left out for simplicity.) This model depicts how each component is utilized and adapts in *retrieval cycles*, which are the periods between one memory retrieval to the next (corresponding to one or more decision cycles):

1. The memory metadata from the previous state persists (M_t).
2. From the previous working-memory state, C_{t-1}, rules fire and modify working memory leading to C_t. At this point, the agent creates a cue Q_t based on the elements in working memory. Technically, the cue is itself part of the context (that is, $Q_t \subseteq C_t$), but we depict it separately in order to match intuition about long-term memory retrieval more closely.
3. Based on the cue, the context, and memory metadata, a memory m_t is retrieved from long-term memory and placed into working memory, adding to the context ($C_t + m_t$). The retrieval leads to changes in memory metadata, which is updated to M_{t+1}.
4. Rules then match on the elements of the retrieved node, and the next retrieval cycle begins.

The importance of such a model of memory is that it clearly lays out the sources of information available; what remains is understanding how long-term memory can use such information to mitigate agent uncertainty. Assuming that a complete model of memory can be built, it may be possible to dispense with the relatively ad-hoc mechanisms discussed in the previous section, and instead develop a well-founded theory of how to globally optimize over multiple sources of information to mitigate uncertainty and incomplete knowledge. Understanding that these mechanism are all address agent uncertainty may provide a path towards a more capable long-term memory as cognitive architectures are used to tackle increasingly complex tasks.

5. Summary and Conclusion

This paper looked at several long-term memory retrieval mechanisms, and showed that they can be framed as mitigating uncertainty during the retrieval process. We note that there are additional memory mechanisms that may be understood in terms of uncertainty which we have not addressed. One such mechanism is *blending* in ACT-R (similar to semantic retrieval in Sigma), in which the retrieved memory is a not one that exists in long-term memory, but a mix of the *features* of existing memories (Lebiere, 1999; Rosenbloom, 2010). This mechanism may be accounting for an unrepresentative sample of memories, while correctly representing the distribution of features. Another mechanism is that of *metamemory judgments*, which provide information about the memory system to the agent for a more strategic use of memory (Li, Derbinsky, & Laird, 2012). Giving the agent more control of the memory system in this way is a concession that, while the agent's rules may not be error-free, the errors and incompleteness in knowledge may be bounded, and it would be foolish to ignore the agent's rules completely.

This paper works towards fulfilling a desire to generally account for uncertainty while leveraging available information for semantic-memory access. For the sake of enabling knowledge-rich agents, this paper has presented a framework for understanding semantic memory, and how its mechanisms are influenced by other components in an agent. We proposed that semantic memory mechanisms are best understood as ongoing attempts to deal with uncertainty, error, and incompleteness in the agent's knowledge, and showed that such a framework leads to satisfactory explanations of multiple existing mechanisms. By explicitly considering the possible sources of uncertainty, we developed a new understanding of spreading activation; by explicitly considering the possible sources of information to mitigate uncertainty, we developed a model which may be used to globally optimize memory retrieval across the mechanisms. Although both of these remain future work, this is a step towards a more complete rational analysis of memory that takes the constraints of the agent into account.

References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review*, *111*, 1036–1060.

Anderson, J. R., & Schooler, L. J. (1991). Reflections of the environment in memory. *Psychological Science*, *2*, 396–408.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., & Hellmann, S. (2009). DBpedia — a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, *7*, 154–165.

Bothell, D. (2013). *ACT-R 6.0 Reference Manual.*

Derbinsky, N., & Laird, J. E. (2009). Efficiently implementing episodic memory. *Proceedings of the 8th International Conference on Case-Based Reasoning* (pp. 403–417). Seattle, WA, USA.

Derbinsky, N., & Laird, J. E. (2010). Extending Soar with dissociated symbolic memories. *Proceedings of the 1st Symposium on Human Memory for Artificial Agents* (pp. 31–37). Leicester, UK.

Derbinsky, N., & Laird, J. E. (2011). A functional analysis of historical memory retrieval bias in the word sense disambiguation task. *Proceedings of the 25th AAAI Conference on Artificial Intelligence* (pp. 663–668). San Francisco, CA, USA.

Derbinsky, N., & Laird, J. E. (2012). Competence-preserving retention of learned knowledge in Soar's working and procedural memories. *Proceedings of the 11th International Conference on Cognitive Modeling*.

Gorski, N. A. (2012). *Learning To Use Memory*. Ph.D. thesis, The University of Michigan.

Laird, J. E. (2012). *The Soar Cognitive Architecture*. MIT Press.

Lebiere, C. (1999). Blending: An ACT-R mechanism for aggregate retrievals. *The 6th Annual ACT-R Workshop*.

Lenat, D. B. (1995). Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, *38*, 33–38.

Li, J., Derbinsky, N., & Laird, J. E. (2012). Functional interactions between memory and recognition judgments. *Proceedings of the 26th AAAI Conference on Artificial Intelligence* (pp. 228–234).

Li, J., & Laird, J. E. (2015). Spontaneous retrieval from long-term memory for a cognitive architecture. *Proceedings of the 29th AAAI Conference on Artificial Intelligence*.

Miller, G. A. (1995). WordNet: A lexical database for English. *Communications of the ACM*, *38*, 39–41.

Nuxoll, A. M., Laird, J. E., & James, M. R. (2004). Comprehensive working memory activation in Soar. *Proceedings of the 6th International Conference on Cognitive Modeling*.

Pirolli, P. (2005). Rational analyses of information foraging on the web. *Cognitive Science*, *29*, 343–373.

Rosenbloom, P. S. (2010). Combining procedural and declarative knowledge in a graphical architecture. *Proceedings of the 10th International Conference on Cognitive Modeling* (pp. 205–210).

Rosenbloom, P. S. (2012). Towards a 50 msec cognitive cycle in a graphical architecture. *Proceedings of the 11th International Conference on Cognitive Modeling* (pp. 305–310).

Thomson, R., & Lebiere, C. (2013). A balanced Hebbian algorithm for associative learning in ACT-R. *Proceedings of the 12th International Conference on Cognitive Modeling*.