Available online at www.sciencedirect.com

SciVerse ScienceDirect

Cognitive Systems Research 24 (2013) 104–113

Cognitive Systems Research

www.elsevier.com/locate/cogsys

Effective and efficient forgetting of learned knowledge in Soar's working and procedural memories

Action editor: David Peebles

Nate Derbinsky *, John E. Laird

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

Available online 5 January 2013

Abstract

Effective management of learned knowledge is a challenge when modeling human-level behavior within complex, temporally extended tasks. This work evaluates one approach to this problem: forgetting knowledge that is not in active use (as determined by base-level activation) and can likely be reconstructed if it becomes relevant. We apply this model to the working and procedural memories of Soar. When evaluated in simulated, robotic exploration and a competitive, multi-player game, these policies improve model reactivity and scaling while maintaining reasoning competence. To support these policies for real-time modeling, we also present and evaluate a novel algorithm to efficiently forget items from large memory stores while preserving base-level fidelity.

Keywords: Large-scale cognitive modeling; Working memory; Procedural memory; Cognitive architecture; Soar

1. Introduction

Typical cognitive models persist for short periods of time (seconds to a few minutes) and have modest learning requirements. For these models, current cognitive architectures, such as Soar (Laird, 2012) and ACT-R (Anderson et al., 2004), executing on commodity computer systems, are sufficient. However, prior work (e.g. Kennedy & Trafton, 2007) has shown that cognitive models of complex, protracted tasks can accumulate large amounts of knowledge, and that the computational performance of existing architectures degrades as a result.

This issue, where more knowledge can harm problem-solving performance, has been dubbed the *utility* problem, and has been studied in many contexts, such as explanation-based learning (Minton, 1990; Tambe, Newell, & Rosenbloom, 1990), case-based reasoning (Smyth & Keane, 1995; Smyth & Cunningham, 1996), and language learning (Daelemans, van den Bosch, & Zavrel, 1999). Markovitch and Scott (1988) have characterized strategies for dealing with the utility problem in terms of information filters applied at different stages in the problem-solving process. One common strategy that is relevant to cognitive modeling is *selective retention*, or forgetting, of learned knowledge. The benefit of this approach, as opposed to *selective utilization*, is that the agent does not have to expend computational resources at run time to decide whether to utilize knowledge or not, a property that may be crucial for real-time modeling in temporally extended, complex tasks. However, it can be challenging to devise forgetting policies that work well across a variety of problem domains, effectively balancing the task performance of models with reductions in retrieval time and storage requirements of learned knowledge.

In context of this challenge, we present two tasks where effective behavior requires that the model accumulate large amounts of information from the environment, and where over time this learned knowledge overwhelms reasonable computational limits. In response, we present and evaluate novel policies to forget learned knowledge in the working

* Corresponding author.
E-mail addresses: nlderbin@umich.edu (N. Derbinsky), laird@umich.edu (J.E. Laird).

1389-0417/$ - see front matter
http://dx.doi.org/10.1016/j.cogsys.2012.12.003

and procedural memories of Soar. These policies investigate a common hypothesis: it is rational for the architecture to forget a unit of knowledge when there is a high degree of certainty that it is not of use, as calculated by base-level activation (Anderson et al., 2004), and that it can be reconstructed in the future if it becomes relevant. We demonstrate that these task-independent policies improve model reactivity and scaling, while maintaining problem-solving competence. To support these policies for real-time modeling, we also present and evaluate a novel algorithm to efficiently forget items from large memory stores while preserving fidelity of base-level activation.

2. Related work

Previous cognitive-modeling research has investigated forgetting in order to account for human behavior and experimental data. As a prominent example, memory decay has long been a core commitment of the ACT-R theory (Anderson et al., 2004), as it has been shown to account for a class of memory retrieval errors (Anderson, Reder, & Lebiere, 1996). Similarly, research in Soar investigated task-performance effects of forgetting short-term (Chong, 2003) and procedural (Chong, 2004) knowledge. By contrast, the motivation for this work is to discover the degree to which forgetting can support long-term, real-time modeling in complex tasks.

Prior work has demonstrated that there are potential cognitive benefits to using memory decay, such as in task-switching (Altmann & Gray, 2002) and heuristic inference (Schooler & Hertwig, 2005). In this paper, we focus on improving reactivity and scaling.

We extend prior investigations of long-term symbolic learning in Soar (Kennedy & Trafton, 2007), where the source of learning was internal problem solving. In this paper, the evaluation domains accumulate information from interaction with an external environment.

Prior work has addressed many of the computational challenges associated with retrieving a single memory according to the base-level activation (BLA) model (Petrov, 2006; Derbinsky, Laird, & Smith, 2010; Derbinsky & Laird, 2011). However, efficiently removing items from memory, while preserving BLA fidelity, presents a different challenge. As such, before presenting the empirical evaluation domains, we formally describe this computational problem; present a novel algorithm to forget according to BLA in large memories; and evaluate our approach with synthetic data.

3. The Soar cognitive architecture

Soar is a cognitive architecture that has been used for developing intelligent agents and modeling human cognition. Historically, one of Soar's main strengths has been its ability to efficiently represent and bring to bear large amounts of symbolic knowledge to solve diverse problems using a variety of methods (Laird, 2012).

Fig. 1 shows the structure of Soar. At the center is a symbolic working memory that represents the agent's current state. It is here that perception, goals, retrievals from

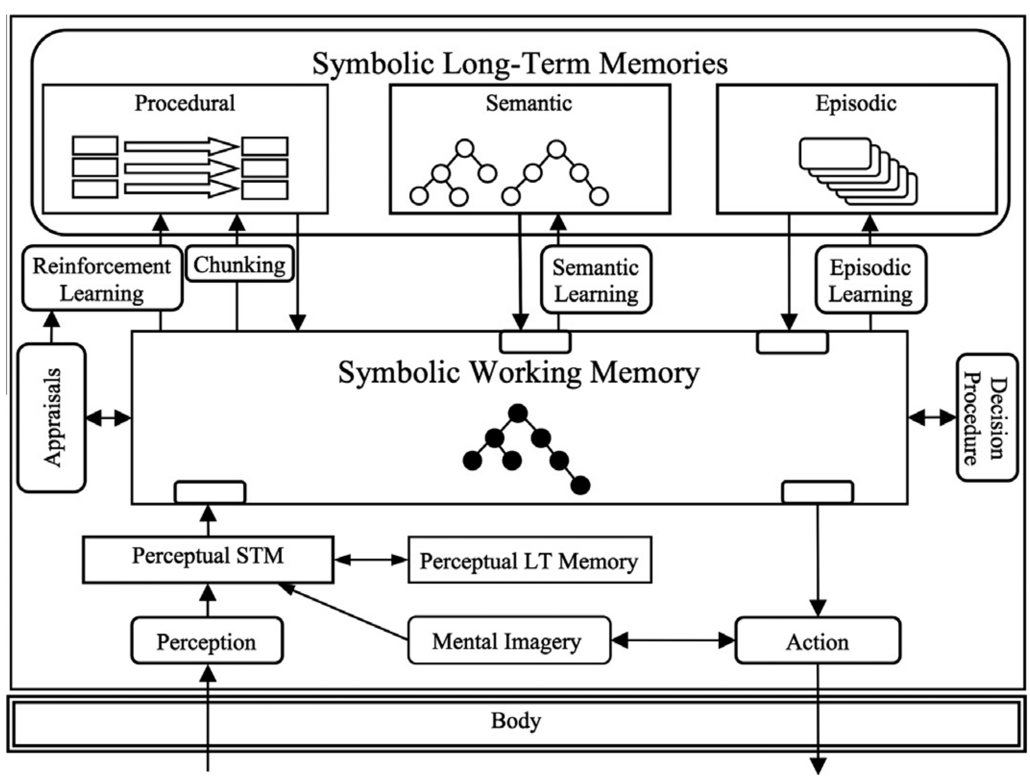

Fig. 1. The Soar cognitive architecture (Laird, 2012).