A Working Memory Simulator for Computational Estimation of Cognitive Load during Learning

François Courtemanche (f.courtemanche@usherbrooke.ca)
Department of Computer Science, Sherbrooke University
Sherbrooke, QC, Canada

André Mayers (andre.mayers@usherbrooke.ca)
Department of Computer Science, Sherbrooke University
Sherbrooke, QC, Canada

Mehdi Najjar (mehdi.najjar@usherbrooke.ca)
Interdisciplinary Research Center on Emerging Technologies, University of Montreal
Montreal, QC, Canada

Abstract
Understanding the cognitive constraints affecting learning is primordial for the effectiveness of instructional design and intelligent learning systems. This paper presents a simulator model for computational working memory load estimations during learning activities. Using a Cognitive Load Theory interpretation, the simulator estimations are used to predict learning outcomes while using a tutoring system.

Keywords: Computational model; working memory; cognitive load theory; cognitive modeling.

Introduction
Nowadays, and thanks to many recent advances in computational technology, intelligent learning systems are increasingly used in order to supplement real classroom teaching. Nevertheless, an important issue concerns the effective learning outcomes of these systems. A growing body of research emphasises the fact that working memory constraints determine instructional effectiveness (Clark, Nguyen & Sweller, 2006; Sweller, van Merrienboer & Paas, 1998). Within intelligent tutoring systems (ITS), few efforts have been made to take into account the improvement of the cognitive aspects related to their usage. Thus, a systematic framework is needed to assess ITS effectiveness from a cognitive point of view. The Cognitive Load Theory (CLT) has been developed to evaluate the impact of the human cognitive architecture on instructional design (Paas, Renkl & Sweller, 2004). Although CLT researches have shown successful results using empirical data, analytical measures and computational models are needed to go a step further in formalising the theory (Paas, Tuovinen, Tabbers, & Van Gerven, 2003). This paper presents a working memory simulator designed to computationally estimate cognitive load. The first section of the article expounds the main architecture of the tutoring system which embeds the simulator. The second and third sections detail the theoretical and computational models of the simulator. Section four (1) explains how computational estimations of working memory load are realised and (2) describes their CLT-based interpretation. Experimental adjustments are given in the last section.

Context
The main role of the working memory simulator is to computationally estimate the mental load undergone by a student while performing problem solving with a tutoring system. This section details the tutoring system architecture and the knowledge representation used to model the student mental operations.

The Intelligent Tutoring System
Similarly to most Intelligent Tutoring Systems (Wenger, 1987), our ITS architecture includes four main modules. Figure 1 shows an overview of the architecture.
The knowledge representation

Our knowledge representation model is implemented as a dynamic graph including three types of memory structure, each one corresponding to a particular type of knowledge. These structures are generally accepted in the computational cognitive modeling literature (Najjar & Mayers, 2007) and represent semantic knowledge, procedural knowledge and episodic knowledge.

Semantic knowledge represents concepts taken in a broad sense where it can be any category of objects. Instances of semantic knowledge contain different slots associated with a particular type (e.g. numeric, string or other concepts).

Procedural knowledge is used to satisfy needs without using the attentional system (Anderson, 1993). In our approach, procedures are subdivided in two main categories: primitive procedures and complex procedures. The former are seen as atomic actions on semantic knowledge and are reified in the system as interactions of the learner via the laboratory interface. The latter represent mental operations and are executed by sequences of actions, which satisfy scripts of goals. Complex procedures results are calculated via functions which computation accounts for processing load (section the simulator section).

Goals are perceived as intentions of the student cognitive system (Najjar & Mayers, 2007; Newell, 1990) and are implemented as generic statements retrieved from the semantic memory (Najjar & Mayers, 2006).

Episodic memory retains details about our experiences and preserves temporal relationships (Tulving, 1983). In our approach, the episodes are based on instantiation of goals. The episodic knowledge is organized according to goals and the procedures needed to achieve them. More precisely, each goal realization is explicitly encoded in an episode.

The automation processes

The two main learning processes defined by the cognitive load theory community are knowledge automation and schema acquisition (Sweller, 1998). In this article, we focus on the modeling of the knowledge automation process, which is applied to our complex procedures.

According to the Instance Theory of Automatization (Logan, 1988), automation reflects a transition from performance based on an initial algorithm to performance based on memory retrieval. The theory suggests that subjects store and retrieve representation of each individual encounter with a stimulus. The transition is explained as a race between the algorithm and the retrieval process governed by statistical processes. The next equation (Logan, 1988) expresses the probability that the algorithm is used.

\[
\text{Probability}_{[\text{algo first}]} = \frac{W_a}{W_a + nW_m} \tag{1}
\]

The algorithm and memory retrieval processes can be viewed as Poisson processes with rates \(W_a\) and \(nW_m\) – where \(W_a\) and \(W_m\) are respectively the mean time for the algorithm and the memory processes and \(n\) is the number of memory traces.

In our system, each procedure realization is encoded in an episode and stored in the episodic memory. The student episodic memory can be used to estimate the automation degree of complex procedures. We adapted the Logan equation in order to take into account the available attentional resources during knowledge encoding. The following equation is used to estimate the complex procedures automation level:

\[
\text{Automation}_{CP} = 1 - \frac{1}{(1 + N)} \tag{2}
\]

\[
N = \sum (S \times G) - \sum (W \times G)
\]

The \(S\) variable represents the number of successful procedure executions encoded in the learner episodic memory. The \(G\) variable is the attentional factor (see the cognitive load estimation section). This variable is added to models the student attentional resources available at encoding time.

The Theoretical Model

The working memory simulator implemented in the tutoring system is inspired by the Baddeley model (Baddeley, 1986; Baddeley, 2003; Repovs & Baddeley, 2006). Figure 2 illustrates its four components which interact to achieve complex cognitive tasks (see Baddeley, 2003).

![Figure 2 - Multicomponent Model of Working Memory](image)

In our system, each procedure realization is encoded in an episode and stored in the episodic memory. The student episodic memory can be used to estimate the automation degree of complex procedures. We adapted the Logan equation in order to take into account the available attentional resources during knowledge encoding. The following equation is used to estimate the complex procedures automation level:

\[
\text{Automation}_{CP} = 1 - \frac{1}{(1 + N)} \tag{2}
\]

\[
N = \sum (S \times G) - \sum (W \times G)
\]

The \(S\) variable represents the number of successful procedure executions encoded in the learner episodic memory. The \(G\) variable is the attentional factor (see the cognitive load estimation section). This variable is added to models the student attentional resources available at encoding time.

The Working Memory Simulator

Our computational implementation of the Baddeley working memory model contains the central executive, the episodic buffer and the visuospatial sketchpad. Due to the nature of
our tutoring system, where no information is communicated to the student in a verbal form, the phonological loop is not incorporated. The student internal verbal information is assumed to be rapidly encoded in the episodic buffer and is manipulated in its semantic form.

**General view**

The semantic and procedural knowledge of the learner are stored in the long-term memory. Concrete instances of this knowledge that must be addressed simultaneously during problem solving are transferred in the working memory (WM). The relation between the knowledge representation and the WM simulator is established via complex procedures. Each time the learner interacts with the lab interface, his/her actions are associated with primitive procedures and are interpreted as parts of complex procedures. The former are seen as interface manipulations and the execution of the latter as mental operations. Thus, the role of the WM simulator is to estimate the mental load resulting from the execution of complex procedures.

Within the cognitive psychology field, working memory generally refers to “the system or mechanism underlying the maintenance of task-relevant information during performance of a cognitive task” (Miyake & Shah, 2002). A number of models investigate working memory under the two functions of storage and processing of information (Anderson, Reder, & Lebiere, 1996; Baddeley, 1986; Cowan, 2005). Following most working memory models, our simulator includes these two processes in order to compute load estimations. As all theories assume a capacity limitation (Cowan, 2001; Miller, 1956), our simulator has a limited capacity on which rely storage and processing. These two components are implemented following a soft trade-off model of resource sharing. The next two subsections describe how processing (equation 3) and storage (equation 8) participate to the working memory load estimation (equation 9) and how they share resources.

**Processing**

Each complex procedure is associated with a function which computes its result. To formalise the concept of function, we took up the definition of Halford (1998), where a function is defined as a mental process over arguments. The processing complexity of a complex procedure is calculated as the number of interacting variables that must be represented in parallel to perform the task (Halford, 1998). In our computational model – which the simulator is based on – the central executive is responsible for computing functions and arguments are represented as instances of semantic knowledge.

A number of studies indicate that the maximum number of interacting variables that can be processed simultaneously is approximately four (Cowan, 2001; Halford, 1998). Thus, the central executive processing capacity limit is attained with execution of quaternary functions. This limit leads to the following equation for the processing demand factor for complex procedures (CP).

\[
\text{Processing factor} = \frac{\text{Arguments}}{4} \quad (3)
\]

Following the ACT-R activation distribution principle (Anderson, 1993; Anderson, Reder & Lebiere, 1996), the central executive resources are equally distributed over the function arguments.

**Storage**

The resource sharing between processing and storage is expressed by the arguments accessibility within complex procedures. Each argument has an access probability depending on its functional location. In our model, there are three possible modes for argument access: computation, long-term memory retrieval and visual access.

Arguments may be available via other functions, as in the case of intermediate results or conversions. In this case, they are associated with the image of the related function. According to the Instance Theory of Automatization (Logan, 1988), this situation leads to two possible access modes: (1) computation and (2) long-term memory retrieval.

**Computation**

In the case where the procedure computing the function argument is not automated, the semantic knowledge instance is created and stored in the episodic buffer as suggested by Baddeley, (2003). Retrieval probability for the episodic buffer depends on the two main theoretical accounts of information loss in working memory: (1) interference and (2) decay.

(1) The interference model developed by Oberauer & Kliewig, (2006) states that items in working memory interfere with each other through interactions of their features. During dual task execution, interference results from the representational overlap between the processing and storage stimuli. When different items share features, loss can occur while they compete for the same feature activation. In our model, interference affects the probability of correctly recalling an item from the episodic buffer. The proportional feature loss effect is estimated with the following equation defined by Oberauer & Kliewig (2006):

\[
\text{Interference} = (1 - C/2)^{n-1} \quad (4)
\]

were \(n\) corresponds to the number of knowledge instances maintained in working memory (buffer and processing) and \(C\) to the average proportion of overwritten features between the item under retrieval and the \(n-1\) others in WM. The latter parameter is calculated regarding the different slots type of the semantic knowledge instances maintained in WM.

(2) The decay effect for memory retrieval within the episodic buffer follows researches on time-based resource sharing. According to Barouillet, Bernardin & Camos (2004), decay is a function of the time during which the concurrent processing totally captures attention and thus impedes refreshing. Memory traces to be recalled suffer from decay as soon as attention is switched away. The effect on maintenance is especially detrimental when concurrent tasks involve memory retrievals. In order to take into
account the effect of the concurrent tasks processing complexity, we modified the decay equation proposed by Barouillet et al. (2004):

\[
\text{Decay} = \frac{\sum a_i r_i + \sum c_j p_j}{T}
\]

(5)

where \(r_i\) corresponds to concurrent retrievals and \(a_i\) to their relative difficulty. The \(a_i\) parameter is estimated with equation 6. The \(c_j\) parameter stands for the number of concurrent computations in the central executive between the moment the item is stored in the episodic buffer and the beginning of the retrieval process. \(p_j\) represents the processing complexity of these computations (equation 3). \(T\) is the total duration of the current procedure. The latter \(c_j\) and \(p_j\) parameters account for the trade-off effect between executive processing and storage in the episodic buffer.

The access probability of knowledge instances stored in the episodic buffer (EB) after a computation process is then computed with the next equation:

\[
\text{Access}_{\text{EB}} = \frac{1}{D}
\]

(6)

where \(I\) and \(D\) correspond to the interference and decay phenomena.

**LTM retrieval** In the case where the complex procedure (CP) computing a function argument is automated, the associated knowledge instance is directly retrieved from the long-term memory instead of being calculated. The following equation accounts for the access probability resulting from LTM retrieval:

\[
\text{Access}_{\text{LTM}} = 0.9 \times A_{\text{CP}}
\]

(7)

Variable \(A\) – representing the procedure automation level – is computed with equation 2 in order to bind LTM retrieval cost with automation strength. As indicated by Logan (2004), storage occurring by means of LTM retrieval has low impact on processing. The 0.9 constant accounts for the low resource sharing effect of LTM access.

**Visual Access** When information is visually accessible in the lab interface, the related knowledge instances are stored in the visuospatial sketchpad (VS). In the actual stage of our tutoring system, visual information is always accessible. Thus, the VS maintenance mechanisms are not incorporated in the simulator. Instances of knowledge visually accessible via the lab interface are stored in the VS visual cache as defined by Logie (1995). The related access probability is 100% and has no trade-off effect with processing.

Following the last three access modes for complex procedure (CP) arguments, the storage factor – which accounts for the effect of information maintenance on working memory load – is computed with the next equation:

\[
\text{Storage factor}_{\text{CP}} = \frac{1}{\prod \text{Access prob.}_{\text{arg}}}
\]

(8)

The access probability for each function argument depends on its functional location (episodic buffer, LTM retrieval and visual) and their related accessibility equation.

**Overall view**

Following the storage and processing resource sharing hypothesis, the cognitive load estimation associated with the computation of complex procedures within the central executive is estimated using the next equation:

\[
\text{Cognitive Load}_{\text{CP}} = P_f \times S_f
\]

(9)

where \(P_f\) (processing factor) and \(S_f\) (storage factor) are computed with equation 3 and equation 8. As for Halford et al. (1998), when a complex procedure recursively engage other procedures for arguments computation, the most demanding one is taken as the cognitive load critical value.

Figure 3 illustrates the working memory simulator during the computation of a non-automated hexadecimal subtraction procedure with the following problem \(D2A-ACF\). The subtraction’s first column is currently processed in the simulator.

**Cognitive Load Estimation**

This section describes how the working memory load measures computed via the simulator are incorporated under the Cognitive Load Theory (CLT) framework.
The Cognitive Load Theory

The Cognitive Load Theory (CLT) is a framework representing characteristics of the mental effort that results from the performance of complex cognitive tasks during learning (Clark, Nguyen & Sweller, 2006; Paas, Renkl & Sweller, 2004; Sweller, van MerrienBoer & Paas 1998). CLT is based on the interaction between knowledge structures and human cognitive architecture and aims to identify optimal methods of instruction.

Cognitive load is a construct defined by three components: intrinsic cognitive load (ICL), extraneous cognitive load (ECL) and germane cognitive load (GCL). ICL represents the load imposed on working memory by the intrinsic nature of the knowledge to be learned. It is affected by the number of elements to be addressed simultaneously in WM. This number depends in turn on the knowledge aggregation degree (schemas structure) in long-term memory – information stored in long-term memory is organised in the form of schemas in order to bind together elements of knowledge handled to achieve common goals. ECL represents all form of load which is not directly devoted to the execution of the current task (e.g. visual and auditory presentation, material manipulation). This load is not effective for learning and can be reduced by a better instructional design. GCL represents the load resulting from learning processes (automation, schema acquisition). It is a form of extraneous load that actively participates in learning. These three types of cognitive load are additives. Their sum may not exceed WM capacity without severely impairing learning or causing a failure of the ongoing task. Thus, the aim of the cognitive load theory is to elaborate instructional settings which balance these three types of cognitive load in order to optimise learning.

Cognitive Load Patterns

In order to optimise instructional settings, the tutoring system pedagogical agent needs information about the mental load resulting from the execution of the different learning tasks proposed to a student. More precisely, it needs to know how much germane cognitive load is available for the student during learning. In this sense the role of a CLT-based learner agent is to provide patterns representing the cognitive load composition (intrinsic, extraneous and germane) undergone by the student at each resolution step when solving a given learning task.

Intrinsic cognitive load (ICL) – which results from the knowledge manipulation in the working memory – is estimated with the working memory simulator. At each resolution step during problem solving, the WM load for the related complex procedure is estimated with equation 9 and is taken into account for ICL value. Depending on the student expertise level and its prior knowledge, the different arguments accessibility varies and provides personalised cognitive load estimation. For instance, during complex calculations, students having automated a large number of simple calculation procedures access intermediate results via LTM instead of maintaining them in the episodic buffer. During learning, ICL measures for a same complex procedure will decrease in function of the progressive knowledge automation due to its utilisation.

The tutoring system interface agent provides extraneous cognitive load (ECL) estimations by analysing the manipulations executed by the student while performing procedures via the lab interface. Since this paper focuses on working memory simulation resulting from knowledge manipulation, ECL estimation is not discussed here.

A number of researches indicate that knowledge encoding processes require more attentional resources than retrieval processes (Craik, Govoni, Naveh-Benjamin & Anderson, 1996; Naveh-Benjamin, Craik, Perretta & Tonev, 2000; Naveh-Benjamin, Guez & Sorek, 2007). In addition, Craik et al. (1996) demonstrated that memory performance tends to be more sensitive to disruptive tasks at encoding than at retrieval. Recall may be autonomous and draw on cognitive resources while encoding is under subject’s control and suffers from low attention. Following these researches, in our model the germane cognitive load (GCL) part – which represents the attentional resources dedicated to fostering learning processes – is attributed the reminder of the total resources available as illustrated by the next equation:

\[ GCL = X - (ICL + ECL) \] (10)

where \( X \) corresponds to a cognitive constant expressing the working memory capacity limit (see next section). Craik et al. (1996) demonstrate that during procedure processing by the central executive, argument accesses always have the priority over knowledge automation (encoding). Therefore, the computed GCL value accounts for the attentional factor during procedure automation – the \( G \) variable in equation 2 – and directly affects learning outcomes. As mentioned by Paas et al. (2003), this germane load analytic estimation allows a better distinction of the differentiated effect of the three types of cognitive load during learning activities.

Empirical Calibration

Experimentations have been made in order to calibrate the cognitive constant \( X \) (equation 10) – which account for the working memory capacity limit.

Cognitive load researchers mostly rely on rating scales in order to evaluate the student mental effort (Paas et al., 2003). These subjective techniques are based on the assumption that subjects are able to introspect on their cognitive processes and report the amount of mental effort expended (Gopher & Braune, 1984). A number of researches (cf. Paas, 1992; Paas & van MerrienBoer, 1994) have shown that people have no difficulty in assigning numerical values to the imposed mental load and that reliable measures can be obtained with unidimensional rating scales. In this sense, we estimate the \( X \) constant using a 7-point rating scale (cf. Kalyuga, Chandler & Sweller, 1999; Pollock, Chandler & Sweller, 2002) where categories ranged from extremely low mental effort (1) to extremely high mental effort (7), accounting for overall mental load. Each analytic cognitive load measure estimated with the
WM simulator is associated with its related empirical value.
The $X$ constant is then calibrated against the maximum subjective load measure bearable by the student without causing a failure of the ongoing task. This alignment can be calculated for each student using the tutoring system in order to individualise the model.

Conclusions
We have proposed a simulator model for computational working memory load estimations. The simulator is based on the Baddeley multicomponent model of working memory. The model has been empirically calibrated. More experimental validations and possible extents are planned. These aspects will be discussed in future papers.

References