# An Algorithm for Simulating Human Selective Attention

Giovanna Broccia[1], Paolo Milazzo[1](✉), and Peter Csaba Ölveczky[2]

[1] Department of Computer Science, University of Pisa, Pisa, Italy
{giovanna.broccia,milazzo}@di.unipi.it

[2] Department of Informatics, University of Oslo, Oslo, Norway
peterol@ifi.uio.no

**Abstract.** The brain mechanism of *selective attention* plays a key role in determining the success of a human's interaction with a device. If the user has to perform concurrent tasks by interacting simultaneously with more than one device, her/his attention is directed at one of the devices at a time. Attention can therefore be seen as a shared resource, and the attentional mechanisms play the role of a task scheduler. In this paper we propose an algorithm for simulating the human selective attention. Simulations can then be used to study situations in which a user has to interact simultaneously with multiple devices. This kind of study is particularly important in safety-critical contexts in which failures in the main task, such as driving a car or setting an infusion pump, may have serious consequences.

**Keywords:** Simulation algorithm · Human-computer interaction
Selective attention · Cognitive load

## 1 Introduction

A key goal of interface design is to make it easy for the user to perform the tasks at hand (e.g., driving a car or withdrawing money) by interacting with a device (such as the car driver interface or an ATM). Good interface design therefore requires understanding how the user perceives and interprets the state of the device, recognizes the enabled actions, memorizes information, and makes decisions based on such information. These cognitive processes can be modeled [5], and techniques such as simulation and model checking can be applied to analyze and predict users' behaviors.

Reasoning about users' behaviors is nontrivial even with one task, such as withdrawing money from an ATM, and becomes an even more complex problem when the user has to perform multiple tasks concurrently. In particular, it is very hard to reason analytically about how the user distributes attention to the different tasks. Analyzing attention is particularly important in order to predict the behavior of users involved in concurrent tasks.


A. Cerone and M. Roveri (Eds.): SEFM 2017 Workshops, LNCS 10729, pp. 48–55, 2018.
https://doi.org/10.1007/978-3-319-74781-1_4

*Working memory* is among the cognitive resources that are mostly involved in interactions with computers and other technological devices. It is a volatile memory used to store and process the information necessary for performing a task. Several models of the working memory have been proposed in the psychological literature, based on different hypotheses about the structure and functioning of such a cognitive system [2,3,6,8,10]. These models all agree on the central role of (selective) attention in the regulation of the working memory activity.

According to some psychological studies [9], the *cognitive load* of each task (i.e., the amount of cognitive resources each task requires) influences the activity of the attentional mechanisms. In particular, focusing attention on a "main" task (such as driving a car) may be impeded by a secondary "distractor" task (such as finding an interesting radio show) with a high cognitive load.

Another factor influencing attention is the fact that some tasks (e.g. driving a car or setting an infusion pump) might be more critical than others (e.g. setting the address in a satellite navigator or resizing the window of the virtual clinical folder application). If the user is involved in different concurrent tasks, one of which is safety-critical and the others non-critical but characterized by a high cognitive load, such a cognitive load of the non-critical tasks could cause the attention of the user to be moved away from the safety-critical task.

We propose an algorithm that allows us to simulate the human selective attention of users involved in multiple concurrent tasks, some of which may be safety-critical. Simulations allow us to get a quick feedback about whether a human can safely perform multiple such tasks, or about which changes should be made to the interface of a device to make interacting with it not too distracting from another (possibily critical) task. We also show that the proposed algorithm is consistent with the description of human selective attention in the psychological literature.

## 2 Cognitive Load and its Influence on Selective Attention

The *cognitive load* of a task is a measure of the amount of the user's cognitive resources required for completing the task. For example, solving a sudoku puzzle is a task with a high cognitive load, since the player has to repeatedly perform difficult computations. On the other hand, chatting with a friend on a social networking website has a lower cognitive load since the actions to be perfomed are easier and less frequent (one has to wait for the friend to reply).

The main cognitive resource used during the execution of a task is the *working memory (WM)* [1,12]. The WM is a form of memory with limited capacity that is responsible for the transient holding and processing of information. It is involved in accomplishing cognitive activities such as reasoning, decision making, learning and problem solving [7].

Different models have been proposed in the literature to explain how the WM works [2,3,6,8,10]. Although these models are based on different hypotheses, they all agree on two important aspects of the WM: it can store a limited amount of items (that dacay over time; i.e., are quickly forgotten) and it is responsible for both processing and storage activities. The limited capacity of the WM is thought to be the cause of the phenomenon known as the processing-storage trade-off: under heavy memory load, resources that are devoted to storage are no longer available for processing, and performance deteriorates.

There are several hypotheses regarding the nature of the items decay. One is that memory traces in WM decay within a few seconds, unless refreshed through rehearsal, and because the speed of rehearsal is limited, we can maintain only a limited amount of information [14]. The theory most successful in explaining experimental data on the interaction of maintenance and processing in WM is the "Time-Based Resource Sharing Model" [4]. This theory assumes that items in WM decay unless they are refreshed, and that refreshing them requires an attentional mechanism.

The attentional mechanism is also needed for any processing task executed concurrently with memory refreshment, especially when the processing components require retrieval from long-term memory. Both processing and maintenance of information in the WM therefore share the same resource: the attention. When there are small time intervals in which the processing task does not require attention, this time can be used to refresh memory traces. When attention is switched away from the items to be recalled, their activation suffers from a time-related decay. This effect would be particularly pronounced when the processing component involves memory retrieval from long-term memory.

The amount of forgetting therefore depends on the temporal density of attentional demands of the processing task. Such a temporal density is actually the measure of the task cognitive load considered in [4]. It is formalized as a value $CL$ denoting the fraction of the time during which a task totally captures the user's attention, and impedes refreshing decaying memory traces. As this time increases, the pauses during which the attention can be directed at refreshing decaying items become less frequent and shorter.

A task can be seen as a sequence of basic activities, each one requiring the user's attention. Basic activities can be of different types (pressing a button, reading a text, etc.) and difficulties. When such activities are performed at a constant pace, the $CL$ denotes the following value:

$$CL = \sum a_i n_i / T \tag{1}$$

where $n_i$ corresponds to the number of activities of type $i$, $a_i$ is a parameter that represents the difficulty of such activity (i.e., the time during which they totally capture attention), and $T$ is the total duration of the task.

Figure 1 shows a schematic representation of a portion of a WM span task in which the goal of the interaction is to remember a sequence of letters (the ones in the white boxes) while performing some processing activities (e.g., reading aloud some digits or performing an arithmetic operation) that require successive retrievals (gray boxes marked R). The three panels in Fig. 1 illustrate variants of the same task that differ in the amount of cognitive load.

Several studies show how attentional limitations could cause trouble when performing concurrent tasks [9,11,13,15]. In particular, [9] describes the roles

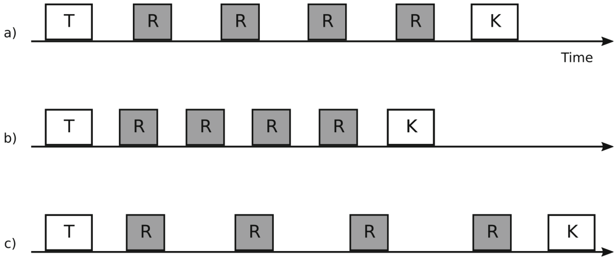


**Fig. 1.** Working memory tasks with different *CL* values: (b) has the highest, (c) has the lowest and (a) has a value between the other two.

of the WM, the *CL*, and the attentional mechanism in the interaction with two concurrent tasks (a "main" task and a "distractor" task). It is shown that when the *CL* of the "distractor" task increases, the interaction with the "main" task could be impeded.

## 3 The Simulation Algorithm

A task can be seen as a sequence of *basic tasks*: single actions that cannot be further decomposed. For example, in the task described in Sect. 2, the basic tasks are actions such as read and store a letter, solve an operation, read aloud a digit, and so on. In the task of sending an email, the basic tasks could be: typing a character, looking for a button in the interface, clicking on the button, etc.

Each basic task is charcterized by a *duration* and a *level of difficulty* (e.g., typing a character is faster and easier than selecting an entry in a dropdown menu). Two consecutive basic tasks may have some idle time between them. Such a time could be necessary for the user to switch from one basic task to the next, but also to the device to process the received input and enable the execution of the next basic task.

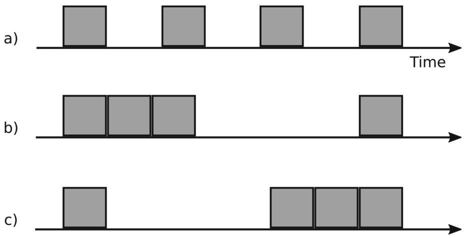


**Fig. 2.** Three different tasks with basic tasks denoted as gray boxes.

In [4], the time between two basic tasks is not explicitly taken into account, since the definition of the *CL* of a task assumes that the single actions are performed at a constant pace (see Eq. 1). According to such a definition, the *CL* of the three tasks depicted in Fig. 2 would be the same. However, if the three tasks in Fig. 2 were potential "distractors" of another "main" task, they would interfere with the main task differently over time: the first one would repeatedly attract the attention of the user, the second one would attract the attention mostly at the very beginning, and the third one mostly after some time. In order to distinguish between these three patterns, the *CL* should not be computed statically as an average value over the whole task duration, but should be computed dynamically as long as the task is performed. In this way, the second task in Fig. 2 would have a higher *CL* at the beginning and a lower *CL* later on, while the third task would have the opposite.

We propose an algorithm for simulating the selective attention in which the *CL* of the concurrent tasks is dynamically recomputed each time a basic task is completed. The *CL* is computed simply as the product of the difficulty and the duration of the next basic task. The *CL* values of the different tasks are then used in order to choose which task to execute next.

The free time between two tasks is modeled as a waiting time before the execution of a basic task (i.e., information is included in the basic task that will follow such a free time). Moreover, each task has a *criticality level* denoting to what degree it is perceived as safety-critical.

**Definition 1.** *A* basic task *is defined as a triple* $\langle w, t, d\rangle \in \mathbb{R}^3$ *where:*

– $w$ *is the* waiting time *before the basic task is enabled;*
– $t$ *is the* duration *of the basic task; and*
– $d$ *is the* difficulty *of the basic task, with* $0 < d \leq 1$.

**Definition 2.** *A* task *is a sequence of basic tasks associated with a criticality level* $C \in \mathbb{R}$ *with* $0 < C \leq 1$.

We represent a task as a pair $[t_1.t_2.....t_n, C]$, with each $t_i$ a basic task, where $\varepsilon$ denotes an empty sequence, and where $C$ is the task's criticality level. Consequently, the pair $[\varepsilon, C]$ represents a *completed task*.

The state of a simulation is given by a *configuration* $\mathcal{C}$, essentialy the set of active tasks, and by a global clock *gc* that will be used to increase the probability of choosing a task that has been ignored for a long time. For this reason, also the timestamps of the last executions of all the tasks are stored in the configuration.

**Definition 3.** *A* configuration $\mathcal{C}$ *is a set of triples* $(tid, T, ts)$ *where:*

– *tid is a* task identifier *(of any type and not repeated in the configuration);*
– $T$ *is a* task $[t_1.t_2.....t_n, C]$*;*
– *ts is a* timestamp *storing the last time the task with identifier tid was executed.*

We define a few auxiliary functions that are used by the simulation algorithm.

Given a task $T$, functions $hd(T)$ and $tl(T)$ give its first basic task and the sequence of the other basic tasks, respectively. Moreover, $criticality(T)$ gives the

**Algorithm 1.** Algorithm for simulating selective attention

1: **while not** $\mathit{completed}(\mathcal{C})$ **do**
2: **if** $\mathit{enabled}(\mathcal{C}, gc) \neq \emptyset$ **then**
3: **for all** $(\mathit{tid}, T, ts) \in \mathit{enabled}(\mathcal{C}, gc)$ **do**
4: $\alpha_{tid} := c \cdot t \cdot d \cdot (1 + (gc - ts))$
5: where $\langle w, t, d \rangle = hd(T)$ and $c = \mathit{criticality}(T)$
6: **end for**
7: choose $(\overline{\mathit{tid}}, \overline{T}, \overline{ts}) \in \mathit{enabled}(\mathcal{C}, gc)$ with probability $\frac{\alpha_{\overline{tid}}}{\sum_{(tid,T,ts) \in enabled(\mathcal{C},gc)} \alpha_{tid}}$
8: $gc := gc + \overline{t}$ where $\langle \overline{w}, \overline{t}, \overline{d} \rangle = hd(\overline{T})$
9: $\mathcal{C} := (\mathcal{C} \setminus (\overline{\mathit{tid}}, \overline{T}, \overline{ts})) \cup (\overline{\mathit{tid}}, tl(\overline{T}), gc)$
10: **else**
11: $gc := \mathit{min}\{ts + w \mid (\mathit{tid}, T, ts) \in \mathcal{C} \wedge \langle w, t, d \rangle = hd(T)\}$
12: **end if**
13: **end while**

criticality level of $T$. Given a configuration $\mathcal{C}$, $\mathit{enabled}(\mathcal{C}, gc)$ gives the set of the tasks that are enabled at time $gc$. A task can be performed if the waiting time of its first basic task has passed since the execution of the previous basic task:

$$\mathit{enabled}(\mathcal{C}, gc) = \{(\mathit{tid}, T, ts) \in \mathcal{C} \mid \langle w, t, d \rangle = hd(T) \wedge gc - ts \geq w\} .$$

Furthermore, $\mathit{completed}(\mathcal{C})$ is true if and only if all tasks in $\mathcal{C}$ are completed:

$$\mathit{completed}(\mathcal{C}) = \forall (\mathit{tid}, T, ts) \in \mathcal{C} . \exists C \in \mathbb{R} . T = [\varepsilon, C] .$$

Our simulation algorithm of selective attention is defined in Algorithm 1. The algorithm performs a main loop that essentially executes one basic task in each iteration. The basic task to be executed is the first basic task of one of the enabled tasks. For each such candidate basic task, an *attention attraction factor* $\alpha_{tid}$ is computed as the product of the criticality level, duration, difficulty and time since the last execution. Each of the candidate basic tasks then has a probability of being chosen that is proportional to $\alpha_{tid}$. Once a basic task has been chosen, it is removed from the configuration and the global clock $gc$ is updated. If the algorithm reaches a configuration in which no task is enabled, the main loop performs an iteration in which only the global clock $gc$ is updated.

In order to show that the proposed simulation algorithm simulates selective attention in accordance with relevant literature, let us first consider, for the sake of simplicity, a variant of the algorithm that does not take the task timestamp into account when computing the attention attraction factor $\alpha_{tid}$. This corresponds to modifying line 4 of the algorithm to $\alpha_{tid} := c \cdot t \cdot d$.

Let us consider two concurrent tasks with the same criticality level and each consisting of $k$ identical basic tasks:

$$T_1 = \langle w_1, t_1, d_1 \rangle . \langle w_1, t_1, d_1 \rangle . \ldots . \langle w_1, t_1, d_1 \rangle$$
$$T_2 = \langle w_2, t_2, d_2 \rangle . \langle w_2, t_2, d_2 \rangle . \ldots . \langle w_2, t_2, d_2 \rangle$$

In order to complete both tasks, the simulation algorithm performs exactly $2k$ steps (where a step represents the execution of a single basic task). Since the two tasks have the same criticality level, the probability of completing task $T_1$ at step $n$, with $k \leq n \leq 2k$, is

$$P(T_1, n) = \left(\frac{t_1 d_1}{t_1 d_1 + t_2 d_2}\right)^k \left(\frac{t_2 d_2}{t_1 d_1 + t_2 d_2}\right)^{(n-k)} \binom{n-1}{n-k}.$$

The expected number of steps necessary to complete task $T_1$ can therefore be given as $E[T_1] = \sum_{n=k}^{2k} P(T_1, n)n$, that corresponds to

$$E[T_1] = \left(\frac{t_1 d_1}{t_1 d_1 + t_2 d_2}\right)^k \sum_{n=k}^{2k} \left(\frac{t_2 d_2}{t_1 d_1 + t_2 d_2}\right)^{(n-k)} \binom{n-1}{n-k} n\,.$$

The formula shows that the expected number of steps for the completion of $T_1$ increases with the difficulty and duration of the basic tasks of $T_2$, namely, it increases when the $CL$ of $T_2$ increases. Hence, the algorithm simulates the switching of attention in agreement with what described in [4,9]. However, since the task timestap is not taken into account, this variant of the algorithm could lead to unrealistic starvation cases (e.g., the algorithm could repeatedly skip a task with very low criticality level and $CL$).

Let us now discuss what changes when the task timestamp is taken into account, namely when line 4 is exactly as in the algorithm definition. Formula $E[T_1]$ becomes more complex since the repeated probabilistic events are no longer independent. However, the structure of the formula remains the same, with a result that is still increasing with the difficulty and duration of the basic tasks of $T_2$ (in agreement with [4,9]). In addition to this, the timestamps tend to favor at each step the task that has not been choosen in the previous step. As a consequence, the regular alternation of $T_1$ and $T_2$ is promoted with, as a result, a reduced variance in the distribution of the number of steps necessary to complete $T_1$. Hence, the use of timestamps reduces the probability of unnatural starvation among the tasks.

## 4 Conclusion

We have proposed a probabilistic algorithm for simulating the human selective attention, based on current knowledge of this cognitive mechanism. The algorithm takes into account the cognitive load and the criticality level of the tasks to be performed. It could be used to simulate the interaction of a user with more than one device. Simulating this kind of situation is particularly interesting when one of the devices is associated to a safety-critical task such as driving a car or using an infusion pump. By mean of simulations it could be possible to identify situations in which the non-critical tasks could represent a too high distraction for the user and could lead to failures in the safety-critical task.

The implementation of the algorithm is available at http://www.di.unipi.it/msvbio/software/AttentionSim.html. In future work, we plan to validate the algorithm against data collected by running some experiments with real users concurrently involved in more than one task.

**Acknowledgements.** This work has been supported by the project "Metodologie informatiche avanzate per l'analisi di dati biomedici (Advanced computational methodologies for the analysis of biomedical data)" funded by the University of Pisa (PRA_2017_44).

## References

1. Atkinson, R.C., Shiffrin, R.M.: Human memory: a proposed system and its control processes. Psychol. Learn. Motiv. **2**, 89–195 (1968)
2. Baddeley, A.: The episodic buffer: a new component of working memory? Trends Cognit. Sci. **4**(11), 417–423 (2000)
3. Baddeley, A.D., Hitch, G.: Working memory. Psychol. Learn. Motiv. **8**, 47–89 (1974)
4. Barrouillet, P., Bernardin, S., Camos, V.: Time constraints and resource sharing in adults' working memory spans. J. Exp. Psychol. Gen. **133**(1), 83 (2004)
5. Cerone, A.: A cognitive framework based on rewriting logic for the analysis of interactive systems. In: De Nicola, R., Kühn, E. (eds.) SEFM 2016. LNCS, vol. 9763, pp. 287–303. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41591-8_20
6. Cowan, N.: Attention and Memory: An Integrated Framework. Oxford University Press, Oxford (1998)
7. Diamond, A.: Executive functions. Annu. Rev. Psychol. **64**, 135–168 (2013)
8. Ericsson, K.A., Kintsch, W.: Long-term working memory. Psychol. Rev. **102**(2), 211 (1995)
9. de Fockert, J.W., Rees, G., Frith, C.D., Lavie, N.: The role of working memory in visual selective attention. Science **291**(5509), 1803–1806 (2001)
10. Just, M.A., Carpenter, P.A.: A capacity theory of comprehension: individual differences in working memory. Psychol. Rev. **99**(1), 122 (1992)
11. Lavie, N., Hirst, A., De Fockert, J.W., Viding, E.: Load theory of selective attention and cognitive control. J. Exp. Psychol. Gen. **133**(3), 339 (2004)
12. Miller, G.A., Galanter, E., Pribram, K.H.: Plans and the Structure of Behavior. Adams Bannister Cox, Eugene (1986)
13. Pashler, H.: Dual-task interference in simple tasks: data and theory. Psychol. Bull. **116**(2), 220 (1994)
14. Towse, J.N., Hitch, G.J., Hutton, U.: On the interpretation of working memory span in adults. Mem. Cognit. **28**(3), 341–348 (2000)
15. Wickens, C.D.: Processing resources and attention. Mult.-task Perform. **1991**, 3–34 (1991)