What Makes Interruptions Disruptive? A Process-Model Account of the Effects of the Problem State Bottleneck on Task Interruption and Resumption

Jelmer P. Borst[1], Niels A. Taatgen[1], & Hedderik van Rijn[2]
[1]Dept. of Artificial Intelligence and [2]Dept. of Psychology
University of Groningen
Groningen, the Netherlands
{j.p.borst, n.a.taatgen, d.h.van.rijn}@rug.nl

ABSTRACT
In this paper we present a computational cognitive model of task interruption and resumption, focusing on the effects of the problem state bottleneck. Previous studies have shown that the disruptiveness of interruptions is for an important part determined by three factors: interruption duration, interrupting-task complexity, and moment of interruption. However, an integrated theory of these effects is still missing. Based on previous research into multitasking, we propose a first step towards such a theory in the form of a process model that attributes these effects to problem state requirements of both the interrupted and the interrupting task. Subsequently, we tested two predictions of this model in two experiments. The experiments confirmed that problem state requirements are an important predictor for the disruptiveness of interruptions. This suggests that interfaces should be designed to a) interrupt users at low-problem state moments and b) maintain the problem state for the user when interrupted.

Author Keywords
Interruptions; multitasking; problem state; working memory; computational model.

ACM Classification Keywords
H.1.2 [Model and principles]: user/machine systems – human information processing.
H.5.2 [Information interfaces and presentation (e.g., HCI)]: user interfaces – theory and methods.

INTRODUCTION

The prevalence of interruptions in our society can hardly be overstated. While writing this first paragraph, I drank a cup of coffee, looked at Facebook, checked my phone for messages, had to leave the building for a fire drill, and have been asked a question by a colleague. This cartoon image – true, however – has been confirmed by a long list of observational studies [15,17,19,21,27,28,31,41]. At the same time, it is also well known that interruptions are disruptive: it takes time to resume a task and more errors are made after an interruption [3,18,20,22,23,26,32,35,45]. Given their widespread occurrence and negative effects on performance, it is crucial to gain a detailed understanding of the cognitive processes involved in task interruption and resumption: only that will enable us to develop software that can diminish the disruptive effects of interruptions.

Previous research has focused on factors that make interruptions disruptive [e.g., interrupting task complexity, duration; 13,14,18,20,22,23,34,35,46], on when users can best be interrupted [1,25,27,32], as well as on how users manage interruptions themselves [7,19,29,30,42,45,46]. To improve our understanding of interruptions, the results of these studies should be integrated into a cognitive theory. Such a theory should not only describe in detail what happens at task suspension and resumption and why some interruptions are more disruptive than others, but it should also be capable of making predictions for new tasks.

One proposal for such a theory is Altmann & Trafton's Memory for Goals theory [2]. Memory-for-goals assumes that each task has an associated task goal with a certain activation level. When a primary task is interrupted, its goal is stored in declarative memory. The activation of this goal decays during the interruption, unless it is rehearsed. When the primary task is resumed after the interruption, its goal has to be retrieved from memory, which takes time. Memory-for-goals theory made the prediction that longer interruptions lead to longer resumption processes, which was confirmed by several studies [23,34,35,45]. However, other interruption effects cannot be easily explained within memory-for-goals theory.

In the current paper we will extend memory-for-goals – and its explanatory power – by not focusing on task goals per se, but on the contents of the *problem state* associated with each task. The problem state contains information that is necessary to perform a task, and has been shown to be an important predictor of interference in multitasking [8,10,11,12,37]. By taking the contents of the problem state

CHI 2015, April 18 - 23 2015, Seoul, Republic of Korea

ACM 978-1-4503-3145-6/15/04…$15.00
http://dx.doi.org/10.1145/2702123.2702156

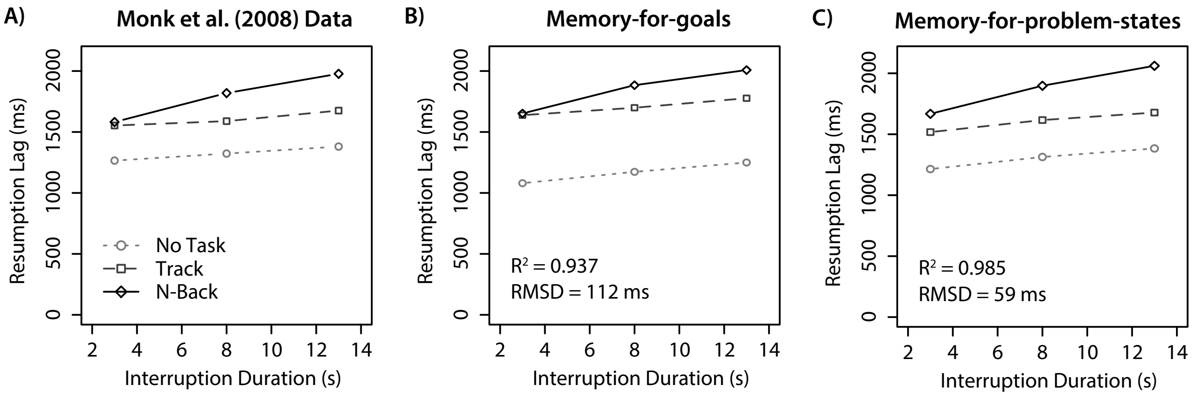

Figure 1. A) Resumption time data of [34], B) model fit of memory-for-goals [43], and C) model fit of memory-for-problem-states.

into account, this new theory, memory-for-problem-states, can explain a wide range of interruption effects. In addition, because memory-for-problem-states theory takes the form of a computational process model, it can be used to derive precise predictions about new combinations of tasks.

The remainder of this paper is organized as follows. First, we will discuss the background literature on interruptions, memory-for-goals, and on the effects of the problem state bottleneck on multitasking performance. We will then give a detailed account of memory-for-problem-states, show that it can account for an existing dataset, and derive two new predictions. Subsequently, we will test these predictions in two behavioral experiments. Finally, we will discuss the implications of our theory for interruption research and derive two recommendations for user interface design.

BACKGROUND

What Makes Interruptions Disruptive?

Researchers have identified several factors that determine the disruptiveness of interruptions. The three most consistent factors are interruption duration, the complexity of the interrupting task, and the moment of interruption.

Several studies have shown that the longer the interruption is, the longer it takes to resume the interrupted task [4,23,35], and the more errors are made after the interruption [4]. In addition, more complex interruptions (e.g., solving 17+36 as compared to 2+3; [23]) also lead to longer resumption times [13,14,20,23,35]. Both effects were nicely demonstrated by Monk et al. [35]. In their study, subjects were asked to perform a fairly complex VCR-programming task. This task was either interrupted by a blank screen, or by one of two different secondary tasks: a tracking task in which subjects had to use the mouse to track an icon on the screen, or a complex n-back task. The length of the interruptions was 3, 8, or 13 seconds. Figure 1A shows how much time it took the subjects to resume the VCR-task after an interruption. Resumption time increased with interruption duration for all three tasks and resumption costs were higher for the more complex interrupting tasks.

A third important determiner of interruption disruptiveness is the moment of the interruption in the primary task. For instance, in a related study to [35], Monk and colleagues again interrupted subjects while they were programming a VCR [34]. Depending on whether subjects were interrupted mid-subtask or between subtasks, interruptions were more or less disruptive. Likewise, Iqbal et al. interrupted subjects either at high or at low workload moments (determined by measuring pupil dilation [24] or by GOMS modeling [26]). These studies confirmed that users take less time to resume a task when interrupted at low-workload moments [25,26] (see [32] for an additional example).

To test whether users are aware that they should switch at low-workload moments, in [42] users were asked to perform a mail-and-internet task, which was interrupted by a chat task. Users were free to respond to the chat messages whenever they thought was best. The results showed that in 94% of the cases users switched at low-workload moments (low working-memory load), showing that users interrupt themselves at the ‘correct’ moments (see also [7,30]). Similarly, [29] showed that users self-interrupted after reducing their working memory load, and [33] that self-interruptions lead to lower interruption costs than forced interruptions (but see [29] for the opposite effect).

In summary, longer and more complex interruptions lead to higher resumption costs, especially if they interrupt the primary task at high workload moments.

Memory for Goals

Memory-for-goals theory explains these effects by assuming that each task has an associated task goal, which is stored in declarative memory when an interruption occurs [2,3]. Each task goal has an activation level, which decays when the goal is not in use. To resume the primary task after an interruption, its goal has to be retrieved from memory. Because the retrieval time of a goal is inversely related to its activation value, the longer the interruption, the more the goal’s activation level will have decayed, and the longer it takes to retrieve it – explaining the interruption duration effect. To account for the effect of interrupting-task complexity, memory-for-goals assumes that users rehearse the primary task goal during an interruption to maintain its activation level. If the interrupting task is more complex, there is less time to rehearse, resulting in higher resumption costs.

In [43] and [44], Salvucci and colleagues used these mechanisms to model the VCR-data of Monk et al. [35]. Figure 1B shows the results: the model matched the data closely. For this model, Salvucci et al. had to make two important assumptions. First, the no-task condition can be resumed fastest because users do not have to reorient after the interruption. Second, users rehearse the primary task goal only for the first 2.5 seconds of the interruption, as opposed to rehearsing it during the whole interruption [44]. Otherwise, there would have been no interruption duration effect in the no-task condition: the activation of the VCR-goal would still be high at the end of the interruption due to rehearsal, independent of the length of the interruption. In the n-back condition there were fewer rehearsals than in the other conditions, because the cognitive system had to spend considerable time on the n-back task itself. As a result, the activation level of the VCR-goal was lower in the n-back condition, which therefore took more time to resume.

Although the explanation of the interruption duration effect in memory-for-goals is straightforward and accounted for several datasets [4,23,35], the explanation for the effect of interrupting-task complexity is less clear. In the example above, it had to be assumed that users rehearsed for a fixed 2.5 seconds – otherwise the model could not explain the effects in the no-task condition. A similar no-task effect was found by Hodgetts and Jones [23], who therefore concluded that it was unlikely that their subjects rehearsed at all. Furthermore, Cades et al. compared resumption times in the VCR task after a simple shadowing-task interruption, after a complex 1-back task, and after a 3-back task [13]. Although resumption was fastest after the shadowing task, there was no difference between the two n-back conditions – while it seems likely that it is harder to rehearse during 3-back than during 1-back. Finally, [29] did not find any evidence for rehearsal during interruptions in pupil dilation data, although it is well known that memory retrievals are a cause of pupil dilation [e.g., 40].

Taken together, memory-for-goals provides a simple and powerful explanation for the interruption duration effect, but its explanation for the interrupting-task complexity effect is less convincing, given that users often do not seem to rehearse during an interruption. Also, it is unclear how memory-for-goals accounts for the moment-of-interruption effect. To improve the explanatory power of memory-for-goals, we will not focus on task goals, but on the contents of the problem state associated with each task.

The Problem State Bottleneck in Multitasking

To perform a task, it is often necessary to maintain some information mentally. For instance, when solving ‘$4x - 8 = 20$’, one needs to store the intermediate solution ‘$4x = 28$’ before calculating the solution. This information is stored in the *problem state*, the central part of working memory that is accessible without a time cost ([5,10], comparable to the *focus of attention* in other theories, e.g., [38]). The problem state acts as a processing bottleneck in multitasking, as it can only maintain information for a single task at a time, and thus causes interference when required by multiple tasks concurrently. In a series of experiments it was shown that subjects were considerably slower and made more errors when the problem state was required for two tasks, as compared to one or none of the tasks [8,10,11,12,37]. Based on these experiments, a cognitive model was developed that gave a close account of the results. In the next section we will combine this model with the ideas behind memory-for-goals to explain task interruption and resumption.

MEMORY FOR PROBLEM STATES

The main idea of memory-for-problem-states is that both the primary and the secondary, interrupting task can have an associated problem state. If both tasks require a problem state, the problem state of the primary task is automatically stored in declarative memory during an interruption, where it starts to decay (analogously to task goals in memory-for-goals). After the interruption, the primary task’s problem state has to be retrieved from memory, which takes time and can lead to errors when an incorrect (i.e. older) problem state is retrieved. As in memory-for-goals, the longer the interruption, the further the activation level of the problem state will have decayed, and the longer it takes to retrieve it from memory – explaining the interruption duration effect.

However, in contrast to memory-for-goals theory, this only occurs when both tasks require a problem state. If the primary task does not require a problem state, there is no information that can decay in memory, so there will not be an interruption duration effect – independent of whether the secondary task requires a problem state. Furthermore, if the secondary task does not require a problem state, it will not disrupt the problem state of the primary task, which will therefore again not show an interruption duration effect. This latter situation explains the effect of interrupting-task complexity: the secondary task will only disrupt the primary task’s problem state if it is sufficiently complex to require its own problem state. Interestingly, memory-for-problem-states therefore predicts that it does not matter how complex the secondary task is, as long as it requires a problem state. This would for instance explain the results of Cades et al. [13], who found that although a no-task interruption was easiest, there was no difference between a complex 1-back and a 3-back interruption.

Implementation Details

Memory-for-problem-states theory was implemented in the cognitive architecture ACT-R [6], to enable quantitative predictions of reaction times and errors. This computational model is a precise implementation of our theory, and can be downloaded from http://www.jelmerborst.nl/models. The main components of memory-for-problem-states are ACT-R’s problem state resource and declarative memory store. As explained above, the problem state resource is used to maintain intermediate information in a task and can store at

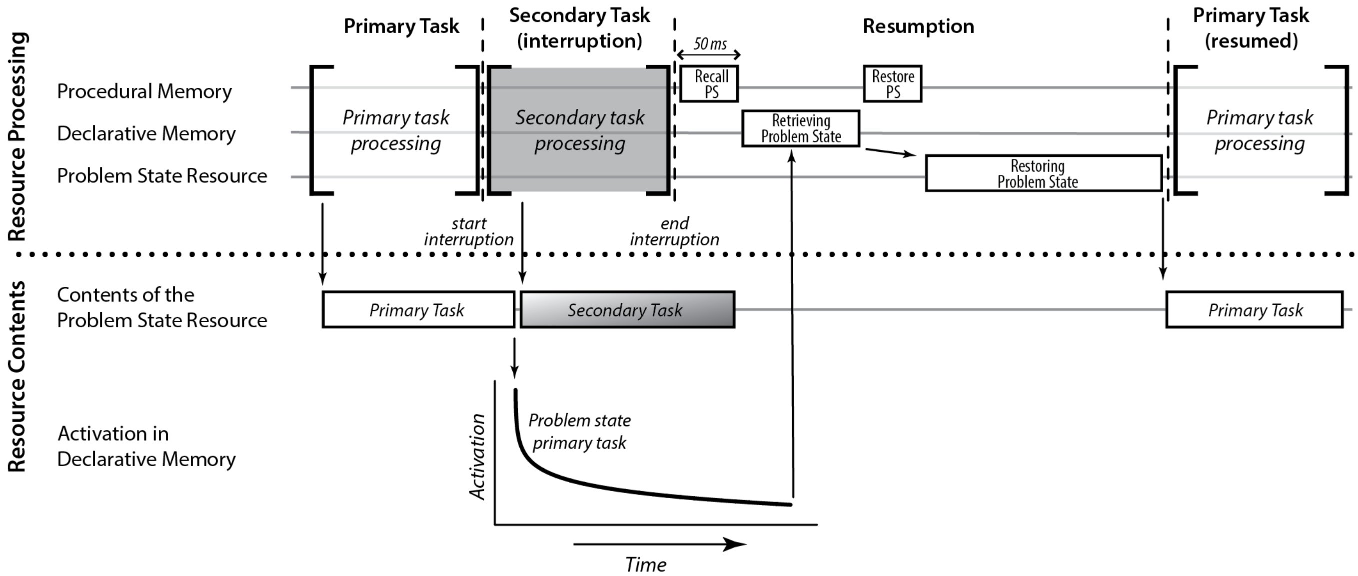

Figure 2. Interruption processing in memory-for-problem-states. Before the interruption, the primary task creates a problem state. This problem state is moved to declarative memory by the secondary, interrupting task, which requires its own problem state. After the interruption the resumption process restores the problem state of the primary task, after which the primary task can be resumed.

most a single chunk of information at a time. It is assumed that information in the problem state can be accessed instantly, without incurring a time cost. On the other hand, it was estimated that it takes 200 ms to store a new representation in the problem state resource [5,10]. When a new representation is stored, previous contents are automatically encoded in declarative memory.

Declarative memory simulates short- and long-term storage of facts. In contrast to the problem state, it contains multiple memory items. Each item has an associated activation level, representing the strength of the item in memory. Activation of an item reflects its frequency and recency of use, and decays with a power function [6]. An item that has been used more frequently in the past will have a higher activation level, as will an item that has been used more recently. The activation A_i of an item at time t is given by:

$$A_i(t) = \ln\left(\sum_{k=1}^{n}(t-t_k)^{-0.5}\right) \tag{1}$$

in which $t_1 \ldots t_n$ indicate the times when the item was used. In the case of an interruption, the problem state will be a new item in memory, and the equation can be simplified to:

$$A_i(d) = \ln(d^{-0.5}) \tag{2}$$

in which d is the time since the start of the interruption.

Retrieving an item from memory takes time. The retrieval time depends on the activation A_i of the item:

$$RT(A_i) = Fe^{-A_i} \tag{3}$$

in which F is a scale factor. The higher the activation level of an item, the faster it will be retrieved from memory. In previous work we have set F to 0.3 [8,9,10,11,12], which we will adopt in this paper, treating F as a fixed parameter. Retrieving information from declarative memory is not always successful: it can either fail altogether because the activation of an item is too low, or a similar item with a higher activation level can be retrieved. In the current paper we will not focus on these mechanisms, but they provide a possible explanation for errors after an interruption.

The final component of the models in this paper is ACT-R's procedural memory [6]. In essence, procedural memory consists of a set of *if-then* rules that govern behavior. For example, there might be a simple rule that says '*if* a fact is retrieved from declarative memory *then* store it in the problem state resource'. It takes 50 ms to 'fire' a production rule, and only a single rule can be fired at a time. For more details, including how production rules are learned, see [6].

Modeling Interruptions

Figure 2 illustrates what happens around an interruption according to memory-for-problem-states theory. It shows the situation in which both the primary and the secondary task require a problem state – for example being interrupted by a phone call while writing an article. The primary task stores a problem state before the interruption, for instance the topic of the current sentence. When the interruption occurs and the secondary task stores its own problem state (say, the date of an appointment), the problem state of the primary task is moved to declarative memory, where it starts to decay (bottom of Figure 3). After the interruption the primary task is resumed. However, first the primary task's problem state – the topic of the current sentence – has to be retrieved from declarative memory.

This resumption process is shown in detail. First, a production rule 'Recall PS' fires that notes that the primary task's problem state it not available. It therefore starts a retrieval from declarative memory. This retrieval takes a certain amount of time. When the problem state is retrieved, it is restored to the problem state resource, and the primary task can resume. Thus, the full resumption process takes 2 × 50 ms (production rules) + 200 ms (restoring the problem state) + the retrieval time of the problem state from declarative memory. The exact length depends on the duration of the interruption: the longer the interruption, the further the activation level of the problem state will have decayed, and the longer it will take to retrieve it.

Modeling Monk et al. (2008)

As a first test of memory-for-problem-states, we used it to simulate the VCR-task of Monk et al. [35]. The resulting fit is shown in Figure 1C. The model follows the memory-for-goals models of this task in that it does not attempt to make a faithful simulation of the VCR task [43,44]. Instead, it is assumed that the VCR task requires two problem states, given that it was a complex task with multiple sub-tasks.

In the no-task condition, the interruption did not require a problem state. However, because only a single problem state of the VCR-task can be maintained in the problem state resource, the second problem state of the VCR-task decays in declarative memory during an interruption. Therefore, resumption costs increase even in the no-task condition: the second problem state has to be retrieved after the interruption, resulting in increased resumption costs with longer interruptions.

Although the tracking task did not require a problem state either – there is no intermediate information to maintain in this task – it did result in higher resumption times than the no-task condition. According to the earlier models this is due to the interface of the experiment [43,44]. The tracking task was presented on the right side of the screen, while the VCR was presented on the left. In the no-task condition, participants could stay focused on the VCR during an interruption, but in the tracking condition they had to look away from the VCR. This resulted in increased resumption costs, as subjects had to refocus on the VCR task after an interruption. Note that the slopes of the resumption costs in the no-task and tracking conditions are similar, supporting the idea that the increase is due to the interface.

The n-back task incurred the greatest costs. According to the model, the n-back task required a problem state to keep track of previously presented letters. This resulted in the loss of both problem states of the VCR-task during an interruption, which decayed in declarative memory. After an interruption, both problem states had to be retrieved from declarative memory, resulting in a steeper slope of the resumption costs with interruption duration than in the other conditions, both in the data as well as in the model.

Predictions

The memory-for-problem-states model fit well to the Monk et al. dataset (R^2 and RMSD-values are reported in Figure 1). Whereas the memory-for-goals models attributed the difference between tracking and n-back to differences in rehearsal [43,44], the new model attributed it to different problem state requirements of the two tasks. Although this meant that we did not have to make somewhat ad hoc rehearsal assumptions, we did have to make the assumption that the VCR task requires two problem states.

As a stronger test of memory-for-problem-states, we will now derive two predictions, and test those predictions in two behavioral experiments. First, memory-for-problem-states claims that the interruption duration effect will only occur if both the primary and secondary task require a problem state, or if the primary task requires multiple problem states. Second, the interruption duration effect should be stronger (i.e. a steeper slope) if the primary task requires multiple problem states. Both predictions run counter to memory-for-goals, which predicts an interruption duration effect independent of the properties of the primary and secondary tasks. In addition, memory-for-goals does not predict that tasks with multiple problem states result in a stronger interruption duration effect, unless they also have multiple task goals.

To test these predictions we performed two experiments. In the first experiment we interrupted a subtraction task with an n-back task (with varying interruption durations). Both tasks could be easy (no problem state) or hard (single problem state). Memory-for-problem-states predicts that we should only find the interruption duration effect when both tasks are hard. In the second experiment a text-entry task was interrupted by an n-back task. The text-entry task required two problem states in the hard condition (but only a single task goal) leading to the prediction that resumption costs should be higher than in the first experiment, and also occur when the secondary task is easy.

EXPERIMENT 1: SUBTRACTION AND N-BACK

In Experiment 1 we used multi-column subtraction as the primary task [from 10], which was interrupted by an n-back task (Figure 3). The subtraction task had two conditions: an easy condition in which subjects did not have to carry between columns, and a hard condition in which subjects had to carry in 5 out of 10 columns. Thus, in the hard condition subjects mentally had to keep track of whether a 'carry' was in progress (solved columns were masked to ensure that subjects maintained carries mentally, Figure 3).

In the n-back task a rapid stream of letters was sequentially presented. Each letter was on the screen for 1100 ms, followed by a mask of 233 ms. The easy condition was a 1-back task: subjects had to indicate whether the current letter was the same or different as the previous letter. A response had to be made during the stimulus; no response was required to the first letter. As the mask was very brief this did not require the use of a problem state: subjects simply judged whether the shape was the same before and after the mask. The hard version was a 2-back task: subjects had to judge whether the current letter was the same as two letters

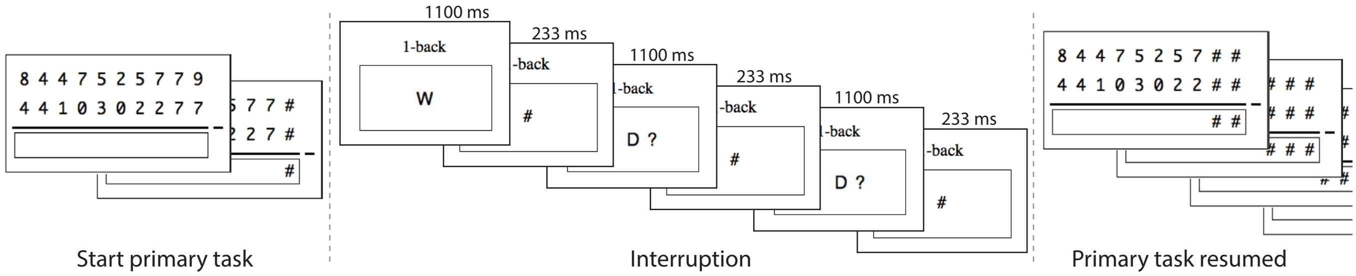

Figure 3. Easy subtraction interrupted by 1-back. The start of a trial is shown, including a 4 second interruption.

back. Thus, in the hard condition subjects needed their problem state to keep track of the letter two-back.

Stimuli for both tasks were generated anew for each subject. Half the responses on the n-back task were 'same' and half 'different'. Additionally, no more than three letters in a row could be the same. Subjects had to press 'x' or 'z' to indicate 'same' or 'different'.

Subjects

39 students of the University of Groningen participated in the experiment for course credit. Three subjects had to be excluded because they underperformed on the hard subtraction task (< 65% correct or slower than 3,500 ms per response), two subjects for underperforming on the hard n-back task (< 80% correct), and one subject because of health problems, leaving 33 complete data sets (23 female, age range 18-27, mean age 20.7). Informed consent as approved by the Ethical Committee Psychology of the University of Groningen was obtained before testing.

Procedure

Each trial started with presenting the conditions for the trial for three seconds, for example: N-Back: easy, Subtraction: hard. This was followed by the subtraction problem, which was interrupted twice by the n-back task. The points of interruption were varied between the 2nd and the 9th column, that is, subjects always started and finished with solving at least two columns. There were also at least two columns between interruptions. This yielded 15 different trial-types: the points of interruption were unpredictable to the subjects. Interruptions consisted of 3, 6, or 9 n-back stimuli, thus 4, 8, or 12 seconds. There was no relation between the length of the first and the second interruption in a trial. Finally, feedback was presented for 3 seconds, indicating the number of correct subtraction columns. Each trial ended with a 4 second fixation screen.

The experiment consisted of a practice block and two experimental blocks. The practice block started with three trials of the easy subtraction task, followed by three trials of the hard subtraction task, seven 6-step trials of the easy n-back task, and seven 7-step trials of the hard n-back task. Then the real task – including interruptions – was practiced in four trials. The two experimental blocks were identical, each consisted of 36 trials: 2 (easy/hard subtraction) x 2 (easy/hard n-back) x 3 (3/6/9 steps first interruption) x 3 (3/6/9 steps second interruption). Trial order was random. The complete experiment consisted of 72 subtraction trials: yielding 144 interruptions per subject, 12 per condition.

Statistical Procedure

A response time in the subtraction task was defined as the time between two responses, or, after an interruption, as the time between the reappearance of the subtraction task and the response. First responses of each trial were removed, as were extreme outliers (RTs <250 ms or >15,000 ms), after which data was removed exceeding three SDs from the mean per condition per subject (2.74% of the data was removed). Analyses are on correct responses only.

Results

Participant accuracy on the n-back task was above 85% in all conditions. Figure 4A shows the resumption costs in the subtraction task. Resumption cost is the extra cost after an interruption as compared to normal responses; it is plotted against n-back interruption duration. The first thing to note is that as long as subtraction was easy, resumption costs did not increase with interruption duration. However, there were fixed resumption costs of about 800 ms in all conditions. In the hard subtraction/easy n-back condition the resumption costs were slightly higher, but did not increase with interruption duration either. Resumption costs only increased with interruption duration in the hard/hard condition. A repeated-measures ANOVA (Table 1) confirmed main effects of Subtraction Difficulty and N-Back Difficulty. Furthermore, the interaction effect between Subtraction Difficulty and N-Back Difficulty was significant, but the three-way interaction between Subtraction Difficulty, N-back Difficulty, and Interruption Duration was not significant.

We subsequently performed simple effects analyses to investigate the model's prediction that we should only observe increasing costs with interruption duration in the hard-hard condition. For each n-back/subtraction condition we tested whether there was an effect of Interruption Duration. These analyses confirmed that there was no effect of Interruption Duration as long as either the subtraction task or the n-back task was easy (Fs < 1). In the hard/hard condition, we found a significant effect of Interruption Duration ($F(1,32) = 4.17$, $p = .0495$, $\eta_p^2 = .12$). Thus, resumption costs only increased with interruption duration in the hard/hard condition, and the ANOVA therefore confirmed the qualitative predictions of our theory.

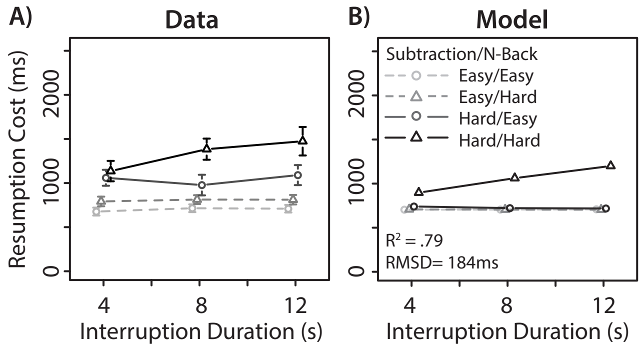

Figure 4. Resumption costs in Experiment 1 (A), and model fit (B). Error bars indicate standard error.

Model

Figure 4B shows the quantitative fit of a memory-for-problem-states model to the data of Experiment 1 (R^2 and RMSD-values are reported in the figure). Memory-for-problem-states predicted that we should only find an effect of interruption duration when both tasks require a problem state. This is clearly shown in Figure 4B, where there is no difference between any of the conditions involving an easy task. The data shows a fixed cost of about 800 ms in all conditions. The model only predicted a fixed cost of ~355 ms. To make the effects comparable, we added 445 ms to the model results in all conditions. This is not meant to give an explanation of these costs, but only to ensure that we estimated the costs due to problem state recovery correctly.

Discussion

In Experiment 1 we tested the prediction that there should only be an effect of interruption duration on resumption costs when both the primary and the secondary task require a problem state. This prediction was confirmed by the data. Our model fit shows that memory-for-problem-states can account for the main effects in the data. However, the model underestimated the resumption costs by about 455 ms in all conditions, and did not predict the overall difference between easy and hard subtraction.

In Experiment 2 we will test the second prediction of memory-for-problem-states: when the primary task requires multiple problem states, resumption costs should be higher, and the resumption costs should increase with interruption duration even if the secondary task does not require a problem state.

EXPERIMENT 2: TEXT-ENTRY & N-BACK

In the second experiment, we used the text-entry task from [10] as the primary task, and again interrupted it with an n-back task, which used digits instead of letters. Unless noted otherwise, the design, procedure and analysis were identical to Experiment 1.

The text-entry task had two conditions: in the easy condition subjects were presented with a letter, they had to click on the corresponding button, followed by the next letter, etc. (Figure 5). In the hard version, subjects had to enter a 10-letter word. This word was shown at the start of a trial, however, as soon as subjects clicked on the first letter the word disappeared, and the rest of the word had to be entered without feedback (subjects did not see what they entered). Thus, in the hard condition subjects had to keep track of what the word was and at which position they were within the word. We assumed that they needed two problem states for this: one for the word and one for the position in the word. Each text-entry trial (entering a 10-letter string) was interrupted twice by the n-back task.

Source	$F(1,32)$	p	η_p^2
Subtraction Difficulty	33.83	< .001	.51
N-Back Difficulty	25.38	< .001	.44
Interruption Duration	3.69	.064	.10
Subtraction x N-Back	5.44	.026	.15
Subtraction x Int. Duration.	1.77	.19	.05
N-Back x Int. Duration	1.83	.19	.05
Sub. x N-Back x Int. Duration	2.51	.12	.07

Table 1. ANOVA results for Experiment 1.

The 10 letter words for the hard version of the text-entry task were handpicked from a list of high-frequency Dutch words to ensure that similarities between words were kept at a minimum. These stimuli were also used in the easy text-entry task, except that the letters within the words were scrambled (under the constraint that a letter never appeared twice in a row). Thus, subjects were presented pseudo-random sequences of letters that they had to enter one-by-one in the easy condition. By scrambling the words, we controlled for letter-based effects, while preventing the use of strategies to predict the next letter.

Subjects

16 students of the University of Groningen participated in the experiment for course credit (12 female, age range 18–22, mean age 19.2). Informed consent as approved by the Ethical Committee Psychology of the University of Groningen was obtained before testing.

Statistical Procedure

Extreme outliers were removed from the data (RTs < 250 ms or > 10,000 ms), after which we removed data exceeding three SDs from the mean per condition per subject (in total 0.92% of the data was removed).

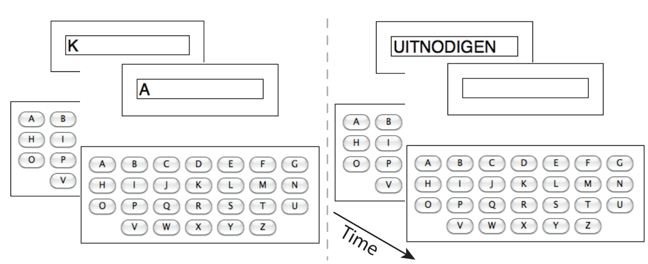

Figure 5. Interface of the easy (left) and hard (right) text-entry task.

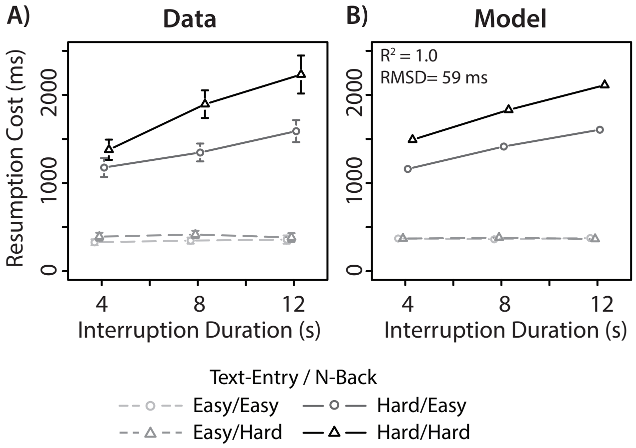

Figure 6. Resumption costs in Experiment 2 (A), and model fit (B). Error bars indicate standard error.

Source	$F(1,15)$	p	η_p^2
Text-Entry Difficulty	155.96	< .001	.91
N-Back Difficulty	18.92	< .001	.56
Interruption Duration	33.11	< .001	.69
Text-Entry x N-Back	16.24	.001	.52
Text-Entry x Int. Duration.	22.04	< .001	.60
N-Back x Int. Duration	3.75	.07	.20
Text-E. x N-Back x Int. Duration	3.18	.09	.17

Table 2. ANOVA results for Experiment 2.

Results

Participant accuracy on the n-back task was in all conditions over 90%, indicating that subjects focused on the n-back task. Figure 6A shows the resumption costs. Again, there was no effect of interruption duration as long as text-entry was easy. However, when text-entry was hard, there was clear interruption duration effect, which was stronger if the n-back task was also hard. The ANOVA (Table 2) confirmed these results: besides main effects of Text-Entry Difficulty, N-Back Difficulty, and Interruption Duration, also the two-way interaction effect between Text-Entry Difficulty and N-Back Difficulty was significant. The three-way interaction showed a trend towards significance.

Model

Figure 6B shows the fit of a memory-for-problem-states model. Because the model assumes that subjects needed two problem states to perform the hard n-back task, it predicted an effect of interruption duration also when n-back was easy, which was confirmed by the data. The effect was even stronger in the hard/hard condition – easy to see in the data (Fig. 6A) but also present in the model (Fig. 6B) – which the model explained by two problem states decaying in declarative memory instead of one, and both problem states having to be restored after an interruption.

As the data shows a fixed cost of about 400 ms in all conditions, we added this cost to the model. Again, this is not meant to give an explanation of these costs, but only to ensure that we estimated the costs due to problem state recovery correctly.

Discussion

Experiment 2 was conducted to test the second prediction of memory-for-problem-states: if a task requires multiple problem states, the resumption costs will be higher and will always increase with interruption duration, also if the secondary task does not require a problem state. This prediction was confirmed, and a memory-for-problem-states model gave a good account of the data. In addition, Experiment 2 confirmed with a new task that there is no effect of interruption duration on resumption costs as long as the primary task does not require a problem state.

GENERAL DISCUSSION

Although interruptions are ubiquitous in our society, an integrated theory of their effects is still absent. In this paper we proposed a first step towards such a theory in the form of a computational process model: memory-for-problem-states. Memory-for-problem-states is based on memory-for-goals theory [2] and on research into the problem state bottleneck [10]. It can be regarded as a more specific version of memory-for-goals: instead of focusing on general task goals, memory-for-problem-states looks in detail at the cognitive requirements of each task. For example, where memory-for-goals predicts an interruption duration effect for every task, memory-for-problem-states specified this to tasks that need a problem state. As support for memory-for-problem-states we first used it to simulate an existing dataset [35]. As a stronger test, we subsequently derived two *a priori* predictions from the model. These predictions were confirmed by two behavioral experiments, yielding strong support for our theory.

Memory-for-problem-states can account for three important factors in interruption disruptiveness: interruption duration, interrupting-task complexity, and moment of interruption. The interruption duration effect is explained by assuming that the problem state(s) associated with the primary task are stored in declarative memory during an interruption, and have to be retrieved on resumption. The longer the interruption, the further their activation in memory will have decayed, and the longer it takes to retrieve them.

The effect of interrupting-task complexity is explained by assuming that complexity can be defined as problem state requirements of the secondary task. The primary task's problem state will only be moved to declarative memory and result in higher resumption costs if the secondary task requires a problem state. Thus, a more complex secondary task that needs a problem state will result in higher costs.

Finally, memory-for-problem-states can explain moment-of-interruption effects. It was shown in multiple studies that it is better to be interrupted between tasks [25,26,34] than in the middle of a task. Memory-for-problem-states explains this by assuming that it is likely that someone has an active problem state in the middle of a task, but not between tasks. During an interruption, this problem state has to be stored in memory, and it has to be retrieved on resumption. For example, in the route-planning task in [24], information had to be maintained as a problem state during a subtask, but not between subtasks. It was shown that this was related to higher workload [24], as well as to higher resumption costs when interrupted during the subtask [25]. This problem-state explanation of the moment-of-interruption effects is also in line with the statistical model of [26], which showed that Task Level, Difficulty of Next Subtask, and Carry-Over between Subtasks, were the most important predictors of interruption disruptiveness. All three factors are clearly related to problem state requirements; especially Carry-Over is an almost identical concept.

Memory-for-problem-states does not explain all costs related to task interruption and resumption. Even in the two relatively simple experiments that we presented in this paper, we had to add a basic cost to the model results to account for the data. These costs might be due to general task reorientation costs, for instance related to task-set reconfiguration (cf. task switching, [36]). In addition, complexity is all-or-nothing in our framework: a secondary task requires a problem state or not. While this is most likely a simplification, it presents a first step to defining complexity in processing terms, which is necessary given that "the term complexity has been inconsistently defined" [35:302] in the literature. Our experiments have shown that problem state requirements are at least an important part of a full complexity definition.

To make interruption less disruptive, memory-for-problem-states makes two suggestions for user interface design. First, our theory suggests interrupting users at low-problem state moments. Where others have successfully interrupted users at low-workload moments [1,24,26,32,34], we specify that low workload should be defined as an absence of problem state usage. As Iqbal et al. have shown, these moments might be determined by using detailed task models [26] or by using physiological measures such as pupil dilation [24]. However, given that pupil dilation is sensitive to many other processes than problem state usage [29,40], task models seem preferable.

Second, our model proposes that resumption costs are due to problem state restoration. To decrease resumption costs, problem state restoration should be aided by presenting the problem state of the user in the environment. The idea of presenting cues to aid resumption is not new [16,33,39]. However, where previous approaches have sometimes recorded the entire task interface [39], we limit this to problem state information. Although this might be difficult – it is information that users maintain mentally, after all – at least in some situations it is possible [e.g., 8].

ACKNOWLEDGMENTS

This research was funded by ERC-StG grant 283597 awarded to Niels Taatgen.

REFERENCES

1. Adamczyk, P.D. and Bailey, B.P. If not now, when? In *Proc. CHI*. 2004, 271–278.
2. Altmann, E.M. and Trafton, J.G. Memory for goals: An activation-based model. *Cognitive Science 26*, (2002), 39–83.
3. Altmann, E.M. and Trafton, J.G. Timecourse of recovery from task interruption: data and a model. *Psychonomic Bulletin & Review 14*, 6 (2007), 1079–84.
4. Altmann, E.M., Trafton, J.G., and Hambrick, D.Z. Momentary interruptions can derail the train of thought. *Journal of Experimental Psychology: General 143*, 1 (2014), 215–226.
5. Anderson, J.R. Human symbol manipulation within an integrated cognitive architecture. *Cognitive Science 29*, (2005), 313–341.
6. Anderson, J.R. *How Can the Human Mind Occur in the Physical Universe?* Oxford University Press, New York, 2007.
7. Bogunovich, P. and Salvucci, D. The effects of time constraints on user behavior for deferrable interruptions. *Proc. CHI*, (2011), 3123–3126.
8. Borst, J.P., Buwalda, T.A., Van Rijn, H., and Taatgen, N.A. Avoiding the Problem State Bottleneck by Strategic use of the Environment. *Acta Psychologica 144*, 2 (2013), 373–379.
9. Borst, J.P. and Taatgen, N.A. The Costs of Multitasking in Threaded Cognition. In *Proc. ICCM*. 2007, 133–138.
10. Borst, J.P., Taatgen, N.A., and Van Rijn, H. The Problem State: A Cognitive Bottleneck in Multitasking. *Journal of Experimental Psychology: Learning, Memory, & Cognition 36*, 2 (2010), 363–382.
11. Borst, J.P., Taatgen, N.A., and Van Rijn, H. Using a Symbolic Process Model as input for Model-Based fMRI Analysis: Locating the Neural Correlates of Problem State Replacements. *NeuroImage 58*, 1 (2011), 137–147.
12. Borst, J.P., Taatgen, N.A., Stocco, A., and Van Rijn, H. The Neural Correlates of Problem States: Testing fMRI Predictions of a Computational Model of Multitasking. *PLoS ONE 5*, 9 (2010), e12966.
13. Cades, D.M., Davis, D.A.B., Trafton, J.G., and Monk, C.A. Does the Difficulty of an Interruption Affect our Ability to Resume? In *Proc. HFES*. 2007, 234–238.
14. Cades, D.M., Werner, N., Boehm-Davis, D.A., Trafton, J.G., and Monk, C.A. Dealing with Interruptions can be Complex, but does Interruption Complexity Matter: A Mental Resources Approach to Quantifying Disruptions. In *Proc. HFES*. 2008, 398–402.
15. Chisholm, C.D., Dornfeld, A.M., Nelson, D.R., and Cordell, W.H. Work interrupted: A comparison of workplace interruptions in emergency departments and primary care offices. *Annals of Emergency Medicine 38*, 2 (2001), 146–151.

16. Cutrell, E., Czerwinski, M., and Horvitz, E. Notification, disruption, and memory: Effects of messaging interruptions on memory and performance. In *Proc. INTERACT*. 2001.
17. Czerwinski, M., Horvitz, E., and Wilhite, S. A diary study of task switching and interruptions. In *Proc. CHI*. 2004, 175–182.
18. Edwards, M.B. and Gronlund, S.D. Task Interruption and its Effects on Memory. *Memory 6*, 6 (1998), 665–687.
19. Eyrolle, H. and Cellier, J.-M. The effects of interruptions in work activity: field and laboratory results. *Applied Ergonomics 31*, 5 (2000), 537–543.
20. Gillie, T. and Broadbent, D.E. What makes interruptions disruptive? A study of length, similarity, and complexity. *Psychological Research 50*, (1989), 243–250.
21. González, V. and Mark, G. Constant, constant, multi-tasking craziness: managing multiple working spheres. In *Proc. CHI*. 2004, 113–120.
22. Gould, S., Brumby, D., and Cox, A.L. What does it mean for an interruption to be relevant? An investigation of relevance as a memory effect. In *Proc. HFES*. 2013, 149–153.
23. Hodgetts, H.M. and Jones, D.M. Interruption of the Tower of London task: support for a goal-activation approach. *Journal of Experimental Psychology: General 135*, 1 (2006), 103–15.
24. Iqbal, S.T., Adamczyk, P., Zheng, X., and Bailey, B.P. Towards an index of opportunity: understanding changes in mental workload during task execution. In *Proc. SIGCHI*. 2005, 311–320.
25. Iqbal, S.T. and Bailey, B.P. Investigating the effectiveness of mental workload as a predictor of opportune moments for interruption. In *Proc. CHI*. 2005, 1489–1492.
26. Iqbal, S.T. and Bailey, B.P. Leveraging characteristics of task structure to predict the cost of interruption. In *Proc. CHI*. 2006, 741–750.
27. Iqbal, S.T. and Horvitz, E. Disruption and recovery of computing tasks. In *Proc. CHI*. 2007, 677–686.
28. Jin, J. and Dabbish, L.A. Self-interruption on the computer: A Typology of Discretionary Task Interleaving. In *Proc. CHI*. 2009, 1799–1808.
29. Katidioti, I., Borst, J.P., and Taatgen, N.A. What Happens When We Switch Tasks: Pupil dilation in multitasking. *Journal of Experimental Psychology: Applied*, (in press).
30. Katidioti, I. and Taatgen, N.A. Choice in Multitasking: How Delays in the Primary Task Turn a Rational Into an Irrational Multitasker. *Human Factors 56*, 4 (2014), 728–736.
31. Mark, G., González, V., and Harris, J. No task left behind?: examining the nature of fragmented work. In *Proc. CHI*. 2005, 321–330.
32. McFarlane, D. Comparison of Four Primary Methods for Coordinating the Interruption of People in Human-Computer Interaction. *Human-Computer Interaction 17*, 1 (2002), 63–139.
33. McFarlane, D.C. and Latorella, K.A. The scope and importance of human interruption in human-computer interaction design. *Human-Computer Interaction 17*, 1 (2002), 1–61.
34. Monk, C.A., Boehm-Davis, D.A., and Trafton, J.G. Recovering From Interruptions: Implications for Driver Distraction Research. *Human Factors 46*, 4 (2004), 650–663.
35. Monk, C.A., Trafton, J.G., and Boehm-Davis, D.A. The effect of interruption duration and demand on resuming suspended goals. *Journal of Experimental Psychology: Applied 14*, 4 (2008), 299–313.
36. Monsell, S. Task switching. *Trends in Cognitive Sciences 7*, 3 (2003), 134–140.
37. Nijboer, M., Borst, J.P., Van Rijn, H., and Taatgen, N.A. Single-task fMRI overlap predicts concurrent multitasking interference. *NeuroImage 100*, (2014), 60–74.
38. Oberauer, K. Design for a working memory. In B.H. Ross, ed., *Psychology of Learning and Motivation*. Academic Press, 2009, 45–100.
39. Renaud, K. Expediting Rapid Recovery from Interruptions by Providing a Visualisation of Application Activity. In *Proc. OZCHI*. 2000, 348–355.
40. Van Rijn, H., Dalenberg, J.R., Borst, J.P., and Sprenger, S.A. Pupil Dilation Co-Varies with Memory Strength of Individual Traces in a Delayed Response Paired-Associate Task. *PLoS ONE 7*, 12 (2012), e51134.
41. Rosen, L.D., Mark Carrier, L., and Cheever, N.A. Facebook and texting made me do it: Media-induced task-switching while studying. *Computers in Human Behavior 29*, 3 (2013), 948–958.
42. Salvucci, D.D. and Bogunovich, P. Multitasking and monotasking: the effects of mental workload on deferred task interruptions. In *Proc. CHI*. ACM Press, 2010, 85–88.
43. Salvucci, D.D., Monk, C.A., and Trafton, J.G. A Process-Model Account of Task Interruption and Resumption: When Does Encoding of the Problem State Occur? In *Proc. HFES*. Human Factors and Ergonomics Society, 2009, 799–803.
44. Salvucci, D.D. and Taatgen, N.A. *The multitasking mind*. Oxford University Press, New York, 2011.
45. Trafton, J.G., Altmann, E.M., Brock, D.P., and Mintz, F.E. Preparing to resume an interrupted task: effects of prospective goal encoding and retrospective rehearsal. *International Journal of Human-Computer Studies 58*, 5 (2003), 583–603.
46. Zijlstra, F.R.H., Roe, R.A., Leonora, A.B., and Krediet, I. Temporal factors in mental work: Effects of interrupted activities. *Journal of Occupational and Organizational Psychology 72*, (1999), 163–185.