A model of free-recall using the ACT-R architecture and the phonological loop

Niels Taatgen
Cognitive Science and Engineering
University of Groningen
Grote Kruisstraat 2/1
9712 TS Groningen
email: niels@tcw2.ppsw.rug.nl

4 July 1996

Abstract

In this paper a model is presented of the free-recall task. It is based on the ACT-R architecture (Anderson, 1993) and a rehearsal theory based on Baddeley's phonological loop (Baddeley, 1986). Rehearsal has been implemented in ACT-R by Nijdam (1995) in our laboratory. Simulation results show that the model can reproduce experimental results reported by Rundus (1971), Postman and Phillips (1965) and Craik (1970). It can offer explanations for the primacy and recency effect, and the fact that the recency effect disappears and becomes negative if there is a delay between presentation and recall.

1 Introduction

Rehearsal, the process of mentally repeating items to facilitate later retrieval, has been studied extensively in seventies in the context of the dual-store memory theory by Atkinson and Shiffrin (1968). In this theory rehearsal plays the important role of the mechanism that is responsible for the transfer of information from short-term memory to long-term memory. One of the experimental tasks used for studying rehearsal is the free-recall task. In this task a list of words, typically containing fifteen to twenty items, is presented at a constant rate to a subject. After presentation, the subject has to recall as many words as possible from the list. Two effects emerge from the results, the primacy effect and the recency effect, respectively referring to the fact that the first and the last few items of the list are recalled better than the rest. The dual-store memory theory can explain both effects: the primacy effect is due to the fact that the first few items in the list are rehearsed more often because they initially don't have to compete for space in STM, and the recency effect is due to the fact that the last few items are still in STM at the moment they have to be recalled. This explanation is confirmed by Rundus (1971), who asked subjects to rehearse aloud. The data show that there is a relation between the number of rehearsals and the chance of recall (figure 1), at least with respect to the primacy effect.

Since the popularity of the dual-store theory declined, partly because rehearsal turned out to be not the sole mechanism to store information in LTM, less research effort has been put into it. Most modern theories about memory do not include rehearsal as a relevant mechanism. A theory that does involve

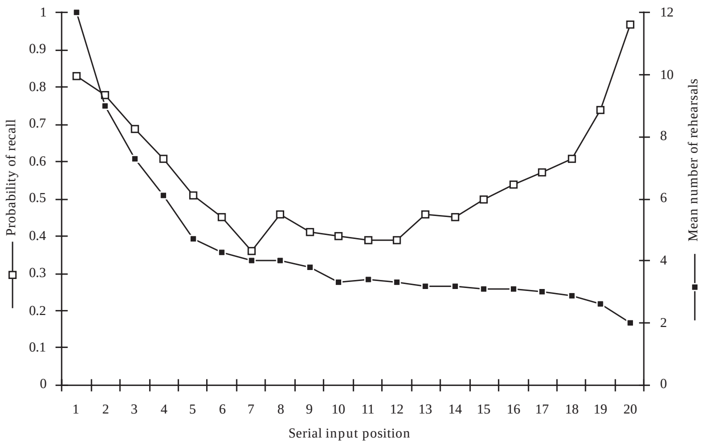


FIGURE 1. The percentage of correctly recalled items and the number of rehearsals (Rundus, 1971).

rehearsal is Baddeley's theory of working memory (Baddeley, 1986). In Baddeley's theory working memory is made out of a central executive and two rehearsal subsystems: the phonological loop and the visuo-spatial sketch pad. Both subsystems are used to temporarily store small amounts of phonological and spatial information. The phonological loop, a system that stores up to two seconds of phonological code in a serial fashion, is responsible for verbal rehearsal, the type of rehearsal most likely relevant for free recall, at least in the overt-rehearsal version by Rundus. In stead of being the process that transfers information from STM to LTM, rehearsal has become a process necessary to maintain items in STM. Whether or not information will also be stored in LTM is not specified by Baddeley's theory, because it is a theory of working memory only. Work by Craik and Lockhart indicates, that the extent in which rehearsed information is stored in LTM depends on the level of processing (Craik & Lockhart, 1972). This led to the distinction between maintenance rehearsal and elaborate rehearsal. Maintenance rehearsal is used just to retain information for a short time, for example a telephone number that needs to be dialled. During elaborate rehearsal on the other hand further processing is done on the rehearsed information.

Baddeley has gathered numerous empirical evidence for the phonological loop and the visuo-spatial sketch pad. The central executive, however, is a weak point in the theory. It is supposed to be able to contain two or three items, and to control what goes into both subsystems, but it is unclear what representation it uses, and why and when it puts something in either subsystem. The central executive is almost a metaphor for the rest of information processing, because it not only stores information, it also makes important decisions on what to memorise in what subsystem. Some of these decisions must be deliberately planned, involving knowledge stored in LTM. The problems with the central executive have an obvious reason: somehow the theory of working memory must be tied to the rest of information processing, and the central executive is responsible for this. Since the rest of information processing is not specified, it is impossible to specify the interface to it.

The ACT-R theory (Anderson, 1993) can be seen as a specification of central information processing that can serve as a means to create models of rehearsal

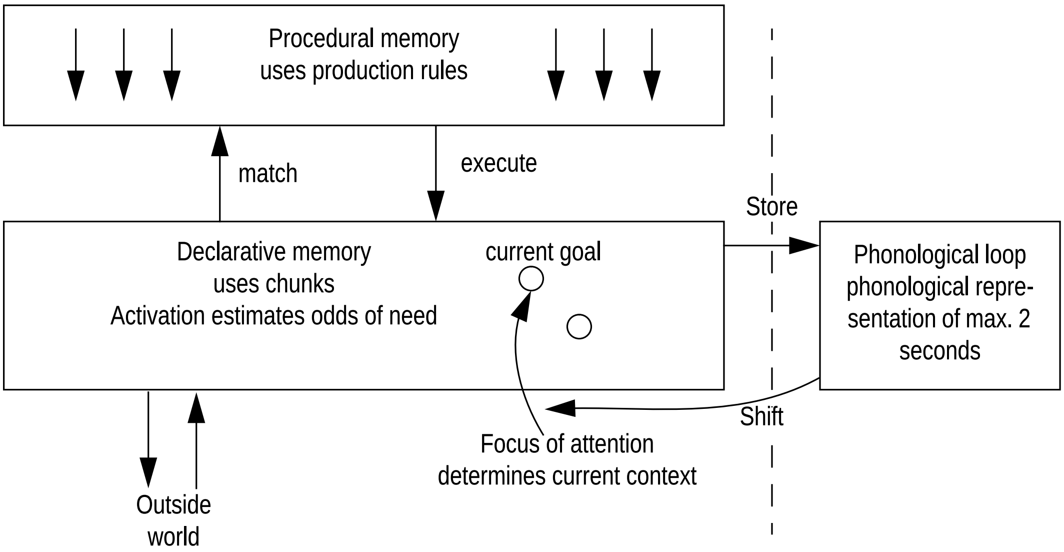


FIGURE 2. Schematic representation of the ACT-R architecture and the proposed addition of a phonological loop.

using Baddeley's phonological loop. The role of the central executive is taken care of by the ACT-R architecture. The ACT-R architecture itself has two memory systems: a declarative and a procedural memory. Declarative memory stores facts called chunks, a frame-like representation, and procedural memory stores procedures, representation by production rules. It has no separate short-term or working memory. To keep track of the current context of the system, a focus-of-attention pointer is used, which points to a single chunk in declarative memory, the current goal (figure 2). Chunks have an activation value, depending on previous use and association values with other chunks. The activation represents an estimate of the odds that the chunk will be needed in the current context. Each time a chunk is retrieved from declarative memory, its activation value is increased. This increase is subject to a decay function, so that the longer ago a chunk has been activated, the lower the contribution to the activation is. The formula to calculate the base-level activation in ACT-R is:

$$B_i = \log \sum_{j=1}^{n} t_j^{-d} + B \qquad \text{(EQ 1)}$$

In this formula, n is the number of times a chunk has been retrieved from memory, and t_i indicates how long ago a particular retrieval was. B and d are constants. Figure 3 shows an example of the behaviour of this function, in which the activation of a chunk is plotted that is accessed at time 1, 4 and 7. ACT-R has several other learning mechanisms: it can learn associations between chunks by association learning, and new productions by analogy learning. For details about all ACT-R learning mechanisms see Anderson (1993).

The activation value of a chunk influences the time to retrieve it. If the activation is very low, the chunk is practically irretrievable. In this sense the activation mechanism of ACT-R can explain the recency effect in free recall: the last

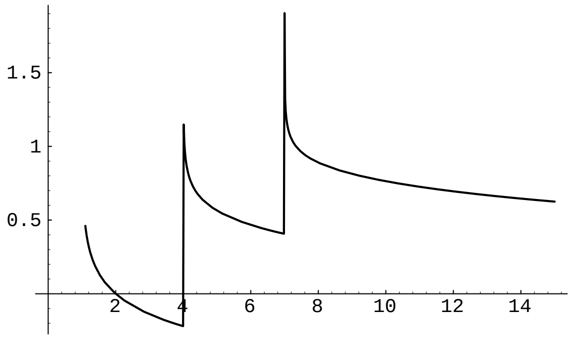


FIGURE 3. Example of the activation for a chunk accessed at time 1, 4 and 7.

few items of the list have been stored more recently, resulting in a higher activation at retrieval time. If we also want to account for the primacy effect, we can use the same explanation as used in the dual-store theory: the first few items are accessed more times at average than the rest of the items, resulting in a higher activation because of a larger *n*.

2 A model of free-recall in ACT-R

To be able to model free-recall in ACT-R, we first need some way to do rehearsal. If want we use Baddeley's phonological loop, we need to make some assumptions about the representation of the loop and the interaction with ACT-R. According to Baddeley, the phonological loop has a phonological representation. To be able to interact with the memory of ACT-R, we must assume a process that translates a phonological representation to a chunk-like symbolic representation in declarative memory and vice-versa.

The impact of rehearsal on long-term memory is dependent on the depth of processing. For the purpose of this article we will distinguish three levels of processing:

1. No further processing at all. Rehearsal is used to just maintain some temporary information with no long-term relevance. In this case rehearsal should have no impact on declarative memory.
2. Rehearsal to retain information that needs to be reproduced at a later time, as is the case in the free-recall task. To increase the chance of retrieval in ACT-R, the activation in declarative memory of the item rehearsed has to be increased.
3. If further elaboration, for example calculation, is to be performed, attention needs to be shifted to the item rehearsed so that appropriate productions may fire.

To accommodate all these levels at the same time in ACT-R, we can shift the focus-of-control between chunks in declarative memory that correspond to items in the phonological loop during rehearsal. At that moment attention is focused on one of the rehearsed items production rules can fire to elaborate the item. If no production rules fire attention is shifted to the next item, which models the first case of rehearsal. If a production fires that simply retrieves the rehearsed item, activation is automatically increased by ACT-R's learning function, producing the second case of rehearsal. If productions fire that do further processing on the retrieved item we have the third case of rehearsal. Any further processing increases activation even more, and also creates opportunities for association learning or even analogy learning.

An implementation in the ACT-R architecture of rehearsal according to the above specification consists of the following elements:

- A data structure to implement a linear phonological storage with a capacity of two seconds. The actual simulation does not use a phonological representation, but uses a symbolic representation with an estimate of the length of the code.
- A function to add an item to the loop. If the capacity of the loop is exceeded, elements are removed from the loop at random until the new item can be accommodated. According to Atkinson and Shiffrin (1968) this procedure corresponds quite well to the behaviour of subjects.
- A function to rehearse the items in the loop. This is implemented by a sub-goal in ACT-R. Executing the rehearsal sub-goal rehearses all the items in the loop by pushing each of them consecutively on the goalstack. A default production just pops each item, implementing maintenance rehearsal (case 1 in the enumeration above). Other task-specific productions can supersede this default production, producing a form of elaborate rehearsal (cases 2 and 3).

The program, implemented by Nijdam (1995) in our laboratory, that implements free recall using rehearsal consists of six production rules:

1. A rule to read or attend to a new word. The new word is added to the phonological loop.
2. A rule to push the rehearsal subgoal during pauses between presentations.
3. A rule to retrieve words from declarative memory during rehearsal, superseding the default rule mentioned above.
4. A rule to start the recall process.
5. A rule to recall a word. Words are recalled only when their activation is above a certain threshold.
6. A rule to end the recall process.

Several variations are possible: the following parameters can be varied easily in the model.

1. Number of words presented.
2. Time between presentation of each word.

3. Whether or not the subjects have to rehearse aloud. Rehearsal aloud is slower than covert rehearsal.
4. Whether or not there is a pause between presentation and recall. During this pause subjects have to do some distracting task like mathematics to prevent further rehearsal (no pause).

3 Results

The free-recall model described above was run 250 times for each simulation, using baselevel-learning in ACT-R with the default parameter settings, except for activation noise, which was set to 0.2. The activation threshold for recall, the minimum activation needed for a successful recall, was set tot 0.0[1]. After 250 runs the data were averaged, resulting in a serial position curve.

Simulation 1

The goal of the first simulation was to reproduce the results of Rundus' experiment. Rundus used 25 subjects, to which 11 lists of 20 words were presented on cards with a 5 second interval. Subjects were instructed to rehearse aloud.

In the experiment the mean number of words correctly recalled was 11.12 and the mean number of rehearsals 88.3. The simulation recalls 11.15 words correctly on average, using 116.0 rehearsals. The serial position curve and the mean number of rehearsals for each item in the list are shown in figure 3. The summed squared error between the data and the simulation is 0.096 for the probability of recall, and 48.4 for the number of rehearsals.

Simulation 2

In the standard free-recall experiment, recall starts immediately after the presentation of the words. If there is a delay between recall and presentation in which further rehearsal is prevented, the recency effect disappears. An experiment by Postman and Phillips (1965) demonstrates this effect: 18 subjects were given lists of 20 words, 6 lists for which recall immediately followed the presentation, and 6 list where subjects had to count backwards for 15 seconds before recall. Words were presented at a rate of one word each second, and rehearsal was covert. The mean number of words recalled correctly was 6.20 if there was no delay after presentation, and 5.05 if there was a 15 second distraction. The serial position curves for both conditions are depicted in figure 4, together with the simulation data. In the condition without delay the data by Murdock (1962) are also shown, since he used the same conditions except that words were read instead of being presented visually. Murdock's subjects recalled 6.87 words on average.

---

[1] Note that in ACT-R activation values can be negative. Moreover, an activation of 0.0 has no special meaning.

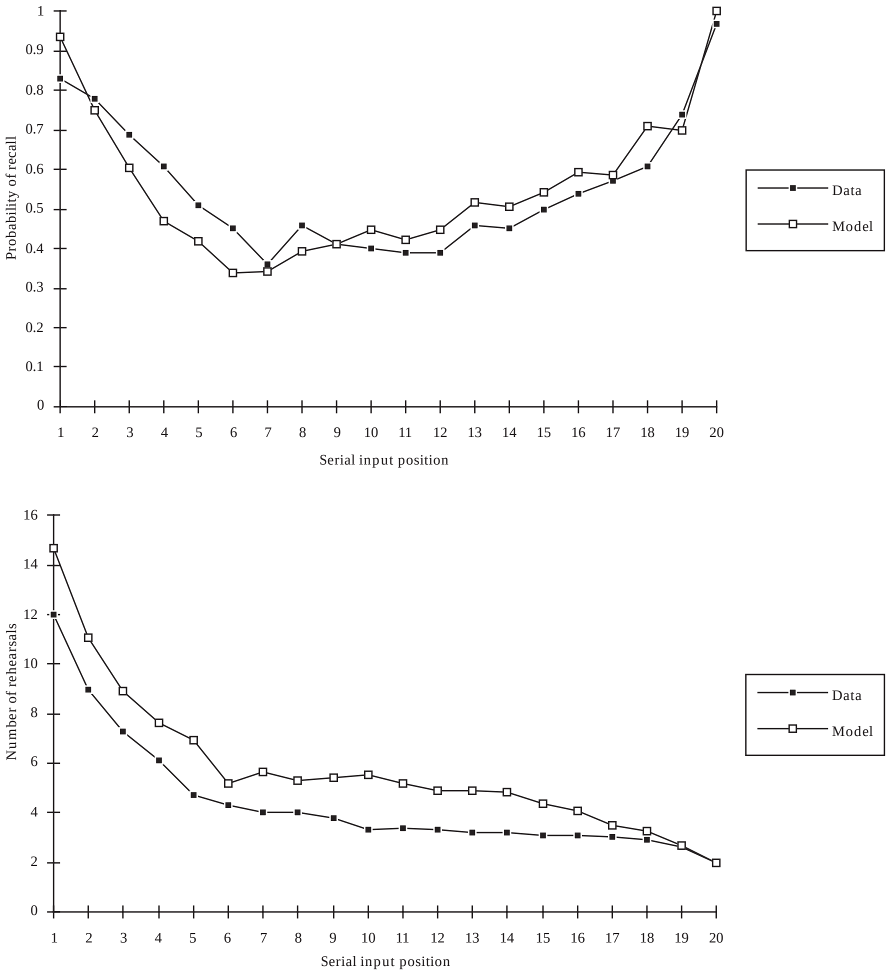


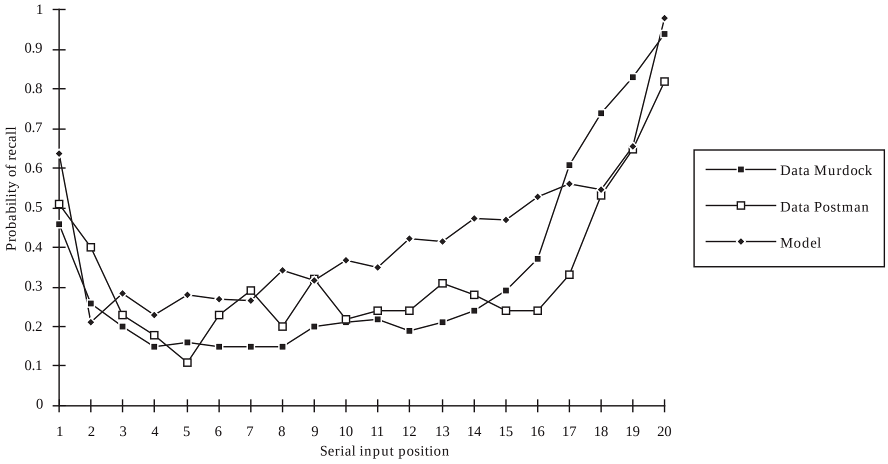
FIGURE 4. Results of the simulation compared to Rundus' data.

The simulation recalls 8.6 words correct on average in the condition without delay, and 4.6 words in the 15 sec delay condition. The summed squared errors can be summarised in the following table:

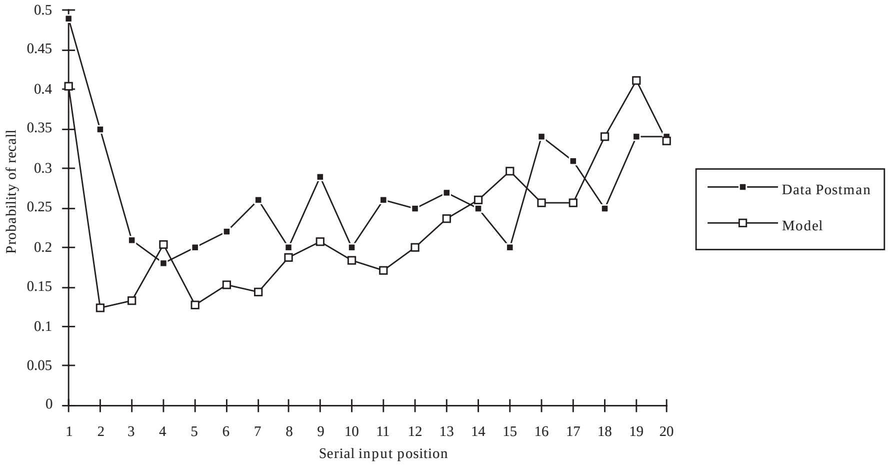
	No delay		15 sec delay
	Murdock data	Model	Model
Postman data	0.27	0.44	0.14
Murdock data		0.46	

Simulation 3

Craik (1970) discovered that the disappearance of the recency effect after a delay can even turn into a negative recency: recall for items at the end of the list is worse than for items in the middle part. In a free-recall experiment 20 subjects were presented 40 lists of 15 words at a rate of 2 seconds per word. After each 10 lists, subjects were asked to recall as much as possible words from the previous 10 lists, giving a final-recall score. The results of this experiment are shown in figure 5a. To obtain a smooth curve Craik averaged each data-point with its successor and predecessor, except for the first and the last.



(a)



(b)

FIGURE 5. Data and model results of free recall without pause (a) and with a 15 second pause (b) after presentation.

The free-recall model also produces negative recency, as can be seen in figure 5b. The same averaging technique is used on the data as Craik used. In the simulation the model has to produce as much items as possible after presentation, after which a 60 second break follows and another, final, recall session. Although the results of the model cannot directly be compared to Craik's data, since the experimental setup is different, a negative recency effect can be seen in the model. The data for immediate recall can be compared directly: the summed squared error is 0.21.

4 Discussion

The results of the simulations show, that the classical effects of primacy and recency in free recall can be reproduced using a theory of rehearsal based on the ACT-R architecture and Baddeley's phonological loop. The primacy effect can be explained by the fact that items early in the list are rehearsed more often on average than other items in the list, the same explanation that was used in

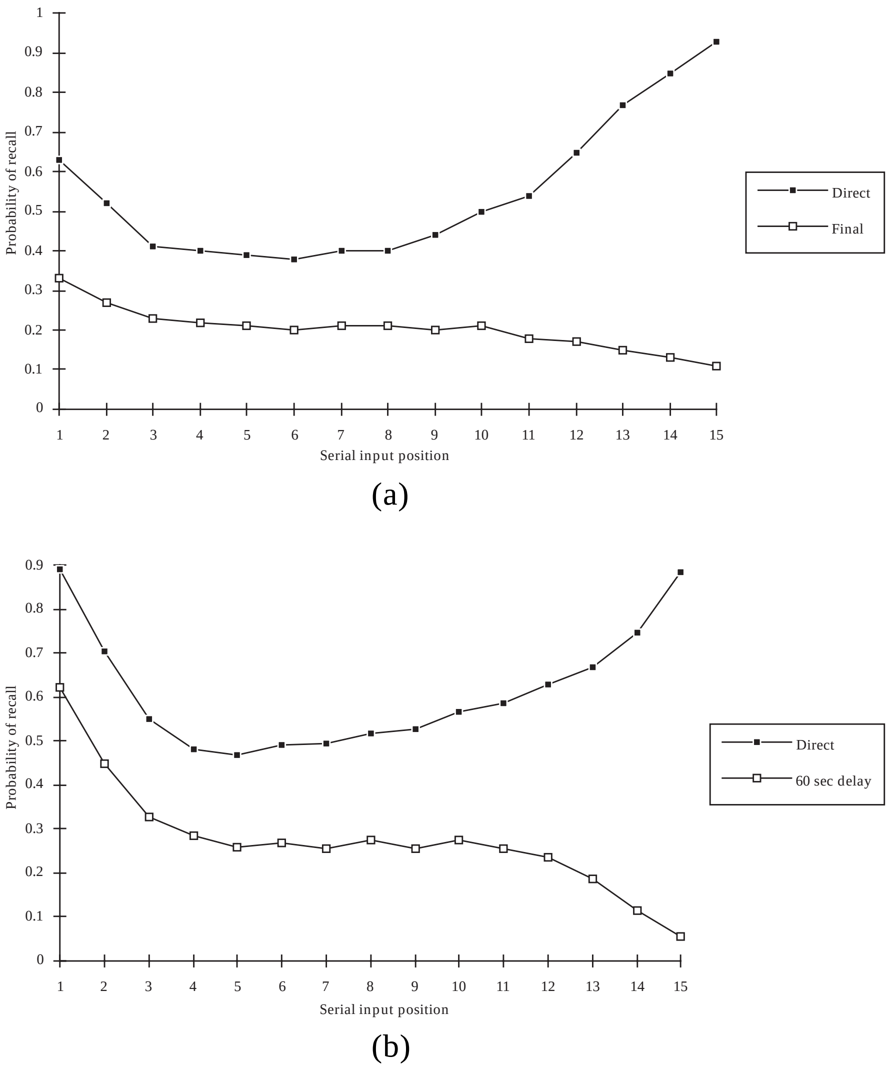


FIGURE 6. Negative recency in data by Craik (1970) (a), and in the model (b).

base-level activation mechanism of ACT-R: the last few items of the list have a higher activation because they have been accessed more recently. Since this activation is subject to an exponential decay function, it decreases rapidly, explaining the fact that the recency effect disappears after 15 seconds. Another property of the decay function is that it becomes more flat as a chunk is accessed more often, due to the summation in equation 1, as can been seen in figure 3. This can explain the fact that the primacy effect is not effected by delays. Rundus' data show, that the last few items in the list are rehearsed least. This can explain why no recency at all eventually turns into negative recency, as was found by Craik.

Since the reports of the experiments that have been simulated contain no information on the variance of individual items in the list, it is difficult to judge how well results from the simulation fit the data. Simulation 1 shows the closest fit between model and data, probably because Rundus' procedure ensures that verbal rehearsal is used to retain the items. In the Craik, Murdock and Postman & Phillips experiments subjects were free to use whatever strategy they liked, for example mental imagery.

One criticism on cognitive models is, that any set of data can be produced by manipulating parameters. To avoid this problem all parameters were set to their recommended default settings, except for the activation noise and activation threshold, which were manipulated to optimise the fit. However, their settings weren't changed between simulations.

It is always a difficult matter to judge whether a cognitive model is a faithful representation of what actually goes on in the mind. One way to look at the matter is to see what explanations a model can offer. The current model casts the old explanation of primacy into a modern framework, and shows that the recency effect can be explained by the memory theory that is part of the ACT-R architecture. The disappearance of recency and the negative recency can be explained by a combination of both.

Further work has to be done on rehearsal, especially in the context of more elaborate rehearsal. Additional exploration of free recall is also possible by simple modifications on the current model.

References

Anderson, J.R. (1993). *Rules of the Mind.* Hillsdale, NJ: Lawrence Erlbaum.

Atkinson, R.C. & Shiffrin, R.M. (1968). Human memory: A proposed system and its control processes. In K.W. Spence & J.T. Spence (Eds.), *The psychology of learning and motivation* New York: Academic Press.

Baddeley, A.D. (1986). *Working Memory.* Oxford: Oxford university press.

Craik, F.I.M. (1970). The fate of primary memory items in free recall. *Journal of verbal learning and verbal behavior, 9,* 143-148.

Craik, F.I.M. & Lockhart, R.S. (1972). Levels of processing: a framework for memory research. *Journal of verbal learning and verbal behavior, 11,* 671-684.

Murdock, B.B. (1962). The serial position effect of free recall. *Journal of experimental psychology, 64*(5), 482-488.

Nijdam, A. (1995). Rehearsal incorporated. Unpublished report, University of Groningen, the Netherlands.

Postman, L. & Phillips, L.W. (1965). Short-term temporal changes in free recall. *Quarterly journal of experimental psychology, 17,* 132-138.

Rundus, D. (1971). Analysis of rehearsal processes in free recall. *Journal of experimental psychology, 89*(1), 63-77.