names arises from the quality of learning, i.e., the task specific constraints that professional driving poses to the taxi drivers, instead of a specific memory architecture or processes. We built this factor into the learning material taking into account what kind routes experienced professional drivers frequently use when navigating in the city of Helsinki.

Besides the explication of the task relevant constraints, another aim of our simulation study was to predict various phenomena observed in expert taxi drivers' performance in a serial recall task. We were especially interested in the phenomena which were assumed to be based on the organisation of environmental knowledge, not on an enhanced rote learning of street names. The taxi driver simulation can account for both qualitative and quantitative results of the memory test. Firstly, differences between the taxi driver model recall scores compared to the novice model are evident in all experimental settings. The taxi driver model retrieves more street names than the novices, and exceed the theoretical short-term memory capacity.

Secondly, the effect of word list order in recall scores was significant only for the taxi driver model: route ordered lists were retrieved better than block-randomised, which in turn were remembered better than random lists. Even in experimental conditions when the list length goes beyond the short-term working memory capacity, the meaningful list organisation facilitated recall. The order of word lists had no influence in the memory performance of novices. They recalled names from lists of length five as poorly as names from lists of length 20, independent of order.

Finally, what are the implications of this study? The aspect that distinguishes our model from previous attempts to simulate expert memory, for instance, EPAM IV (Richman, Staszewski, Simon, 1995) in the chess memory, is that our model learns all its knowledge in the field , i.e., by "driving" in a city. The model is not explicitly trained to memorise lists of street names. By navigating in the modelled city it constructs a complex representation of spatial relations from which it can derive various facts to improve its memory performance.

This is an important distinction because it allows more specific conclusions with respect to the accounts of process and product theories of memory recall. It especially facilitates the analysis of the expert effects which have the potential of improving memory performance in task domains where recall is a contrived task. Particularly, training the model as if memory recall were an intrinsic task leads to better performance because the model is trained with the same material it is expected to remember in the experiment. On the contrary, training the model by task practise, although in a simplified environment, does not bias the memory performance, because the model is not learning to memorise but only accomplish the task.

Acknowledgments

We are grateful for Virpi Kalakoski and Pertti Saariluoma, who allowed us to use their unpublished data for the simulation. We want to thank our excellent students for contributing in the implementation of the model, Saara Huhmarniemi, Mikko Koivisto, Janne Korhonen, Mikko Määttä and Mikko Nikkanen, and Kai Laine for helping to construct the Helsinki City map representation.

References

Chase, W.G. (1983). Spatial representations of taxi drivers. In D.A. Rogers & J.A. Sloboda (Eds.), *The acquisition of symbolic skills* (pp. 391-405). New York: Plenum Press.

Ericsson, K. A., & Kintsch, W. (1995). Long-term working memory. *Psychological Review, 102*, 211-245.

Ericsson, K. A., Patel, V., & Kintsch, W. (2000). How experts adaptations to representative task demands account for the expertise effect in memory recall: Comments on Vicente and Wang (1998). *Psychological Review, 107(3)*, 578-592.

Kalakoski, V. & Saariluoma, P. (in press). Taxi drivers' exceptional memory of street names. *Memory & Cognition*.

Peruch, P., Giraudo, M-D., Gärling, T. (1989). Distance cognition by taxi drivers and the general public. *Journal of Experimental Psychology, 9*, 233-239.

Richman, H.B., Staszewski, J.J., Simon, H.A. (1995). Simulation of expert memory using EPAM IV. *Psychological Review*, Vol. 102, No. 2, 305-330.

Simon, H. A., & Gobet, F. (2000). Expertise effects in memory recall: Comments on Vicente and Wang 1998. *Psychological Review, 107(3)*, 593-600.

Vicente, K. J. & Wang, J. H. (1998). An ecological theory of expertise effects in memory recall. *Psychological Review*, Vol. 105, 33-57.

Vicente, K. J. (2000). Revisiting the constraint attunement hypothesis: A reply to Ericsson, Patel, & Kintsch (2000) and Simon & Gobet (2000). *Psychological Review, 107(3)*, 601-608.

Intention superiority effect: A context-sensitivity account

Christian Lebiere (cl@cmu.edu)
Human Computer Interaction Institute; Carnegie Mellon University
Pittsburgh, PA 15213 USA

Frank J. Lee (fjl@cmu.edu)
Department of Psychology; Carnegie Mellon University
Pittsburgh, PA 15213 USA

Abstract

Intention superiority effect (Goschke & Kuhl, 1993; Marsh, Hicks, & Bink, 1998) is the finding that the times to retrieve memory items related to uncompleted or partially completed intentions are faster than for those with no associated intentions. However, this relationship reverses when the intended tasks are completed (Marsh, Hicks, & Bink, 1998; Marsh, Hicks, & Bryan, 1999). That is, the times to retrieve memory items related to completed intentions are slower than for those with no associated intentions. In this paper, we present a computational account of the intention superiority effect using the ACT-R (Anderson & Lebiere, 1998) cognitive architecture. Our modeling approach is based on the idea that uncompleted or partially completed intentions are available as context in the current goal, and they prime related memory items while inhibiting unrelated memory items. However, once the intended tasks are completed, they are removed from the current goal, which produces an inhibitory effect on memory items associated with them. We describe an ACT-R model that is able to reproduce all of the effects reported in Marsh, Hicks, and Bink (1998).

Keywords: Prospective memory, Intention superiority effect, ACT-R.

Introduction

Prospective memory has recently been receiving a lot of attention among psychologists (Brandimonte, Einstein, & McDaniel, 1996). The interest in prospective memory reflects a trend in psychology to investigate more "real-world" phenomena. For the ACT-R theory (Anderson & Lebiere, 1998), the importance of prospective memory research is clear. First, as a unified theory of cognition, especially with its roots in human memory, the ACT-R theory must endeavor to account for the results from this body of research. Second, as the ACT-R theory is pushed towards more complex and dynamic tasks, an account of prospective memory will be critical, because it is central to planning and multitasking in dynamic task environments.

To begin our task of understanding prospective memory from the ACT-R theoretical framework, we decided to focus on a particular phenomenon in prospective memory called the *intention superiority effect* (Goschke & Kuhl, 1993; Marsh, Hicks, and Bink, 1998; Marsh, Hicks, and Bryan, 1999).

Intention Superiority Effect

Intention superiority effect is the finding that the times to retrieve memory items related to uncompleted or partially completed intentions are faster than for those with no associated intentions (Goschke & Kuhl, 1993; Marsh, Hicks, & Bink, 1998). However, this relationship reverses when the intentions have been completed. That is, the times to retrieve memory items related to completed intentions are slower than for those with no associated intentions (Marsh, Hicks, & Bink, 1998; Marsh, Hicks, and Bryan, 1999). The data reported by Marsh, Hicks, and Bink (1998) provide a good overview of this phenomenon, and we review them here.

Marsh, Hicks, and Bink (1998)

Marsh, Hicks, and Bink (1998) reported results from four experiments, using slight variants of the procedure detailed in Goschke and Kuhl (1993). For each of their experiment, Marsh et al. prepared two pairs of scripts. Each script consisted of a title (e.g. *Setting Table*) and five action propositions (e.g. *set the tablecloth, place the candles*, etc.). The scripts were carefully made so that they were semantically distinct from one another and were counterbalanced. To measure the activation levels of the memory items associated with the scripts, they used response times on lexical decision tasks (LDTs) on the words from the scripts. The main manipulation between the four experiments in Marsh et al. was when the LDTs were given.

In Experiment 1, they had participants memorize a pair of scripts during each block of the two-block experiment. In one of the blocks, participants were told that they would *perform* the tasks specified in one of the scripts, and in the other block, they were told that they would *observe* the experimenter carrying out the tasks specified in one of the scripts to verify that it was performed correctly. Of the two scripts in each block of this experiment, the script that they were told to perform or observe was considered to be the *prospective* script, and the remaining script was considered to be the *neutral* script. The LDTs were given before they performed or observed the prospective script.

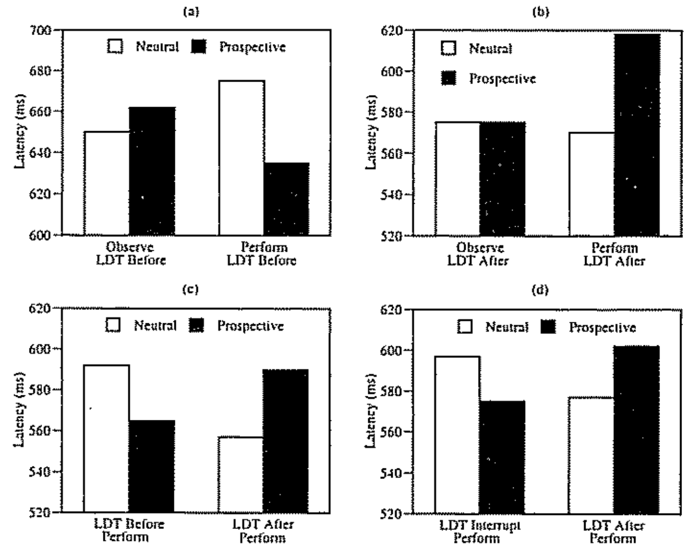

Figure 1: Reproduction of Figures 1 – 4 from Marsh, Hicks, and Binks (1998).

It is important to note that participants were told which script was the prospective script only after they memorized both scripts to criterion. This prevented them from privileged access (e.g., through additional rehearsals) to the prospective script over the neutral script during the initial study phase. After they were told which script was the prospective script, they were immediately given the LDTs.

In Figure 1a, we replot Marsh et al.'s data from Experiment 1. As can be seen, participants were faster in accessing the memory items related to the prospective script compared to their access to the items related to the neutral script. However, note that this difference did not exist for the prospective script that people were told to observe.

Marsh et al.'s Experiment 2 was identical to their Experiment 1, with the exception that the LDTs were given after people performed or observed the prospective script. In Figure 1b, we replot their data from Experiment 2. As can be seen, there is a striking reversal in people's response times to the memory items related to the prospective script after they perform the task. That is, their access to the memory items related to the prospective script was slower after they performed the task compared to their access to the items related to the neutral script in the *perform* condition. However, they found no such difference between the prospective and the neutral script in the *observe* condition.

Since Marsh et al. found no significant differences between the prospective and the neutral script in the *observe* condition in their first two experiments, they decided to focus on the *perform* condition in Experiments 3 and 4. In Experiment 3, the participants were told to perform the prospective script in both blocks of the experiment. In one of the blocks of the experiment, the LDTs were collected before they performed the script, and in the other, the LDTs were collected after they performed the script. In Figure 1c, we replot Marsh et al.'s data from their Experiment 3. As can be seen, they replicated the basic results from their previous two experiments in a within-subjects design. Namely, people were quicker to access the memory items of the prospective script compared to those of the neutral script before they performed the intended task, but after they completed the intended task, they were slower to access the memories items of the prospective script compared to those of the neutral script.

In Experiment 4, Marsh et al. Basically followed the procedure outlined in their previous three experiments, but in one of the blocks of this experiment, they interrupted the participants while they were performing the intended task and gave the LDTs, and in the other block, the LDTs were given after they completed the intended task. The main idea they were testing was the Zeigarnik effect (Butterfield, 1964; Goschke & Kuhl, 1993). The Zeigarnik effect is the finding that people's access to the memory of the task after it is completed is poorer compared to their access to the memory of the task while they are performing it. As Marsh et al. noted, Zeigarnik effect seemed very close in spirit to the intention superiority effect, and hence they decided to investigate it using their experimental paradigm. In Figure 1d, we replot their data from their Experiment 4. As can be seen, the results mirrored those from Experiment 3 and added support for the Zeigarnik effect. Namely, people were quicker to access the memory items related to the prospective script compared to the neutral script during the execution of the intended task, but their access to the memory items related to the prospective script were slower compared to the neutral script after they completed the intended task. This would seem to suggest that the same mechanism underlies both phenomena.

In the next section, we describe an ACT-R (Anderson & Lebiere, 1998) model of the four experiments that we have reviewed above from Marsh et al. (1998).

Model

Symbolic Level

At the symbolic level, the ACT-R model of intention superiority effect, or more specifically of the Lexical Decision Task, is straightforward. There is only one type of declarative knowledge, contained in chunks of type *lexicon*. Those chunks contain three slots: *word*, which holds a word, *spelling*, which holds its spelling, and *context*, which hold the context in which this word occurred. The goal to perform the LDT is also of type *lexicon*. When a goal is completed, it becomes a new memory chunk or reinforces an existing one if an identical chunk already exists in long-term memory. Thus past goals to perform lexical decision tasks become long-term memory structures used in performing future ones.

Table 1: Production rules for Lexical Decision Task.

Name	Production Rules
Map	IF the goal is to perform lexical access on *spelling* and there is a chunk mapping *spelling* to *word* THEN note in the goal that the desired word is *word*
None	IF the goal is to perform lexical access on *spelling* THEN note that no word can be found
Output	IF the goal is to perform lexical access and the word is *word* THEN output *word* and focus on a new goal

Procedural knowledge consists of three production rules. The most important production, **map**, implements lexical access. Given a word's spelling that was encoded from the environment and is present in the current goal, **map** retrieves from declarative memory the lexicon chunk associating that spelling to a word and adds the word to the goal. If the retrieval fails, then the production, **none**, notes in the goal that no word can be found associated to that spelling. After either of these two productions fires, the production, **output**, outputs the word then focuses on a new goal. The English form of these three production rules is given in Table 1. As is the case for the declarative knowledge, these productions are quite simple and are potentially learnable, an important constraint on any model.

Subsymbolic Level

While at the symbolic level the model is appealingly simple and straightforward, it wouldn't generate the prospective memory effects described previously. The symbolic level of productions and chunks merely provides the structure of the model on which the statistical learning mechanisms of the ACT-R architecture operate to tune its performance to the structure of the environment by determining the optimal subsymbolic parameters that control the availability of symbolic structures. The probability and time to retrieve a chunk from declarative memory is a function of its activation, which is given by the activation equation:

$$A_i = B_i + \sum_j W_j \cdot S_{ji} \qquad \text{Activation Equation}$$

A_i is the total activation of chunk i, B_i is its base-level activation, W_j is the attentional level of activation source j and S_{ji} is the strength of association between source j and chunk i. The base-level activation is learned to reflect the context-free history of use of the chunk, with chunks that were used more frequently or recently having higher base-level activation. The activation sources j are the components of the goal, which in the case of the LDT are the context and spelling chunks, evenly dividing between them a total attentional level, W. Therefore the strengths of association are learned to reflect the history of use of a chunk given the composition of the goal. The more a chunk is retrieved when an activation source is present, the larger the strength of association between the two. Formally, the strengths of association between activation sources and chunks reflect the log likelihood ratio of retrieving a chunk given a source over their past history. In addition, activations are stochastic through the addition of zero-mean gaussian noise. We will not describe in detail the equations that control the learning of base-level activations and strengths of associations other than to point out that the parameters controlling that learning as well as the magnitude of the noise were left at the default values used in many other models (Anderson & Lebiere, 1998) and were not optimized to fit the data.

Performance in the retrieval of a chunk is a function of its activation. In this model, we assume that retrieval is assured.[1] We are particularly interested in the latency of retrieval, which is given by the latency equation:

[1] This makes the **none** production unnecessary. However, in the experiment the lexical decision task included some non-words for which no corresponding lexicon chunk would exist. Thus while we did not model that part of the experiment we included here for completeness sake the production to deal with that case.

$$T_i = F \cdot e^{-A_i} \qquad \text{Latency Equation}$$

T_i is the latency to retrieve chunk i and F is a time scaling factor. Thus, the higher the activation of a chunk, the faster its retrieval latency, and vice versa.

Assumptions and Results

The basic assumption of this model lies in the composition of the current goal, which determines the identity of the sources of activation. As previously described, in addition to the essential components of the lexical decision task, namely the spelling and the word to be accessed, the goal also includes a slot that encodes the current context or task to be accomplished. Because one expects the task to be strongly predictive of the words that need to be accessed, as is the case here where each script is associated to a limited number of words, including the task in the goal as a source of activation is a reasonable assumption in trying to maximize the activation of the chunks to be retrieved.

This provides a useful inhibition mechanism.[4] The chunks that were retrieved for a given task will be more active when the task is over, because the base-level learning mechanism has boosted their activation to reflect their recent use. To prevent these chunks from intruding on the following task because of this temporary boost in activation, changing the context to a new task not only boosts the activation of the words most likely to be encountered in that task, but also lowers the activation of the words that are not related to that task, including the words that are temporarily more active due to rehearsal in the previous task.

By boosting performance through the strategic use of basic architectural learning mechanisms, this assumption is therefore compatible with the spirit of the rational analysis underlying the ACT-R architecture (Anderson, 1990). Including in the goal an additional component that generally reflects the current context is similar to the key assumption of an ACT-R model of sequence learning (Lebiere & Wallach, 1998; 2000). In that model, the additional goal component is the previous stimuli in the sequence, but it serves the similar purpose of providing a discriminating source of activation in addition to the primary one, i.e. the current stimuli. This assumption is also compatible with the view of the current goal as ACT-R's working memory (Lovett, Reder & Lebiere, 1999), which makes it a natural place to keep the task(s) to be completed active as a reminder of their impending execution.

The one thing left to specify is which context is active in the goal given the various experimental conditions. The general rule is that if a task is expected to be performed in the near future (and no other pressing one is currently being performed) then the context is set to that task to facilitate the retrieval of related information. When a task has been performed, the context is changed to some other task, even if it is not expected to be performed soon, to prevent the information associated with the task that was just performed from generating excessive interference.

The model worked as follows. Both prospective and neutral scripts were first studied. This means setting the goal context to the title of the script and performing lexical access (through the same productions as used in the lexical decision task) on all the words present in the script (10 for each script). For each word, this typically meant firing the **map** and **output** productions. The retrieval of the lexicon chunk for the word in the map production led to the increasing of the strength of association between the current context and that lexicon chunk. The rest of the operations would vary with each experimental condition. None of the model parameters were optimized to fit the data. The important aspect of the model is how simply it can capture the effects in the data, not the maximization of a quantitative measure of fit.

In Experiment 1, participants engaged in the Lexical Decision Task (LDT) before observing or performing the script, but after having been instructed which script they would have to observe (Observe condition) or perform (Perform condition). In the Perform condition, the context was set to the script to perform, because subjects would have to actively generate the script. In the less demanding Observe condition, the context was randomly set to either script with equal probability, on the assumption that subjects did not care to set the right context because they would merely have to observe the experimenter perform the script, which the externally provided components of the script providing enough spreading activation without needing to make the script itself a source of activation. This reflects the fact that maintaining a context in the goal exacts some costs, including the additional splitting of the total attentional level W. Alternatively, the context could be left empty or set to some other task than those of the experiment, which would yield comparable results.

Figure 2a presents the model results in terms of the average latency to perform the task in each condition. Comparing it to Figure 1a, all the significant effects in the subject data are reproduced. In the Observe condition, neutral and prospective scripts produce similar latencies because the context is equally likely to be set to one or the other, and the experimental conditions prevented one to be studied more than the other. In the Perform condition, the prospective script is recalled faster than in the Observe condition because the context is set to that script, leading through the strengths of association to an activation boost to lexical items present in that script, resulting in lower latency. The neutral script, on the other hand, is slower than both the prospective script and the Observe condition because the context is always set to the other script, leading to a lower activation through negative strengths of associations from the prospective script to neutral script items, and a longer latency.

---

[4] Since formally strengths of associations are log likelihood ratios, unless there is a strong association between a source and a chunk the corresponding probability ratio is often smaller than 1, resulting through the log in negative, i.e. inhibitory, strengths of association. However, there is no fundamental distinction in ACT-R between positive and negative strengths of association.

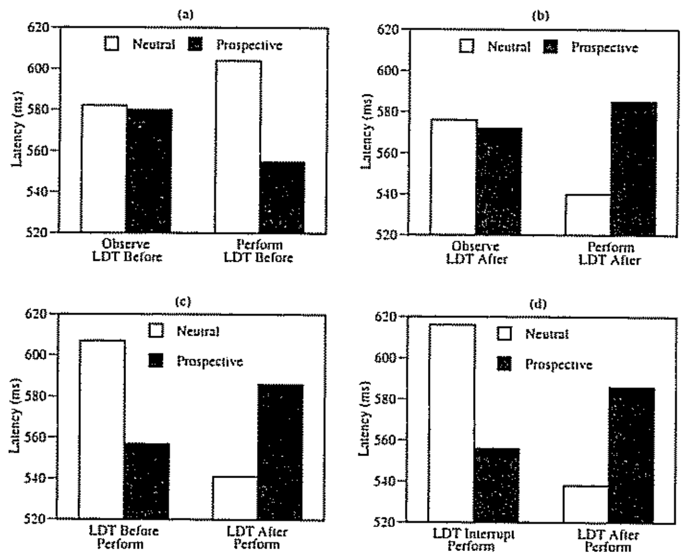

Figure 2: ACT-R Model of the data from Marsh, Hicks, and Binks (1998).

In Experiment 2, the same procedure was followed but the LDT was administered after the subjects observed or performed the prospective script. The observation or performance of the prospective script was modeled as an additional study phase identical to the original one.[3] In accordance with the model setup of Experiment 1, the context in the Observe condition is randomly set to either script while the context in the Perform condition is set to the neutral script to prevent interference from the prospective script. As in the subject data, the model generates roughly equal latency for both scripts in the Observe condition, as shown in Figure 2b, which can be compared to Figure 1b. The reason is that the additional rehearsals of the words in the prospective script have also strengthened the associations to those words from the prospective script. Since half the time the context in the LDT is the neutral script, the advantage of the rehearsal is lessened and access to lexical items associated with the prospective script is only slightly faster. In the Perform condition, the neutral script has much lower latency because the context has always been set to that script, whereas the prospective script, despite its additional rehearsal, has higher latency because of consistent inhibition from the neutral script.

In Experiment 3, only the Perform condition was used, with the LDT administered either before or after the performance of the prospective script. The setting of the context was the same as in the previous experiments, namely the context was set to the prospective script before (and during) performance and to the neutral script afterwards. Figure 2c displays the results, which are consistent with those of the previous experiments, and can be compared to Figure 1c. When the LDT is given before performance, latency is lower for the prospective script because the context is set in its favor. When the LDT is given after the performance, latency is lower for the neutral script, because the context is then set in its favor, and higher for the prospective script. The overall latency is however lower than before the performance thanks to the additional rehearsal.

In Experiment 4, the procedure was similar except that the performance phase was interrupted halfway through and the LDT task administered both during the interruption and after the performance phase had been completed. Again, the context was set according the usual rule, meaning to the prospective script during the performance phase (including the interruption) and to the neutral script afterwards. Figure 2d displays the model results, which again reproduce the effects

---

[3] It might have been reasonable to assume that performance resulted in stronger rehearsal than observation, but there seemed to be no good a priori way of estimating that difference in the model. Thus the model provided equal rehearsal in both conditions.

of the subject data given in Figure 1d. During the interruption, words associated with the prospective script benefits both from the (partial) rehearsal and from the script being in the context. After the performance phase, the switch of the context to the neutral script then favors words associated with that script.

Discussion

Goschke and Kuhl (1993) and Marsh, Hicks and Bink (1998) interpreted their findings in terms of the ACT* cognitive theory (Anderson, 1983), a predecessor of the ACT-R theory, and found their results entirely consistent with the theory. They suggested that intentions were represented as goal nodes, which conferred them additional activation, which was quickly dissipated when the goal was popped off the stack and replaced by another. Marsh, Hicks and Bink (1998) moreover suggested that after completing a task people naturally directed their attention toward making a decision of which task to complete next, providing a rationale for our rule of switching the context to the neutral task after completing the prospective task. They also noted the inhibitory nature of such a switch, which we also observed. While many details have changed between ACT* and ACT-R (most important in this case is that change in activation as a result of goal switching is due to a change in spreading activation rather than a decay in goal activation) the basic account of the data in terms of the ACT theory remains valid.

Another advantage of the model is its compatibility with existing models of language (Anderson, Budiu, & Reder, in press; Lebiere, 1995). As such, this model of the lexical decision task provides a bridge between the written form of words, i.e. their spelling, and their internal representation, in terms of symbolic chunks. An isomorphic model could easily be written for the mapping between auditory input of word components, i.e. phonemes, and the words themselves. In either case, after the mapping between external presentations to internal representations is performed words can be manipulated irrespective of their presentation modality, providing a very desirable abstraction in the form of symbolic chunks. Such abstraction might be one of the fundamental purposes of language.

Conclusion

The main contribution of this paper is to present a simple yet precise model of the intention superiority effect. The model hinges on the fundamental assumption that a task that is or will be accomplished in the near future is kept in the goal as a source of activation, leading to faster access to related lexical items and inhibition of items related to other, competing contexts. However, once the task is completed, it is removed from the goal and attention is switched to a different context, thereby providing inhibition of the just completed task. Additional empirical and modeling work is clearly needed to determine the limits and circumstances of context maintenance. But the fact that such a simple model could provide a precise account of this complex and somewhat surprising phenomenon is an indication of the power of cognitive modeling to illuminate empirical data.

Acknowledgments

The research reported in this paper was supported by the Office of Naval Research, Cognitive Science Program, under Contract Number N00014-95-10223. All correspondences should be addressed to Christian Lebiere at the Human Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

References

Anderson, J. R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.

Anderson, J. R. (1990). *The adaptive character of thought*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Anderson, J.R., Budiu, R., & Reder, L.M. (in press). A theory of sentence memory as part of a general theory of memory. *Journal of Memory and Language*.

Anderson, J.R., & Lebiere, C. (1998). *Atomic components of thought*. Mahwah, NJ: Lawrence Erlbaum Associates.

Brandimonte, M., Einstein, G.O., & McDaniel, M.A. (Eds.) (1996). *Prospective memory: Theory and applications*. Mahwah, NJ: Lawrence Erlbaum Associates.

Butterfield, E.C. (1964). The interruption of task: Methodological, factual, and theoretical issues. *Psychological Bulletin, 62*, 309-322.

Goschke, T., & Kuhl, J. (1993). Representation of intentions: Persisting activation in memory. *Journal of Experimental Psychology: Learning Memory and Cognition, 19*, 1211-1226.

Lebiere, C. (1995). Individual differences in an ACT-R model of sentence reading. Presented at the joint session of the *CAPS Workshop* and the *Second ACT-R Workshop* at Carnegie Mellon University, Pittsburgh, PA.

Lebiere, C. & Wallach, D. (1998). Implicit does not imply procedural: A declarative theory of sequence learning. Paper presented at the *41st Conference of the German Psychological Association*, Dresden, Germany.

Lebiere, C., & Wallach, D. (2000). Sequence learning in the ACT-R cognitive architecture: Empirical analysis of a hybrid model. In Sun, R. & Giles, L. (Eds.) *Sequence Learning: Paradigms, Algorithms, and Applications*. Springer LNCS/LNAI, Germany.

Lovett, M. C., Reder, L. M., & Lebiere, C. (1999). Modeling working memory in a unified architecture: An ACT-R perspective. In Miyake, A. & Shah, P. (Eds.) *Models of Working Memory: Mechanisms of Active Maintenance and Executive Control*. New York: Cambridge University Press.

Marsh, R.L., Hicks, J.L., & Bink, M.L. (1998). Activation of completed, uncompleted, and partially completed intentions. *Journal of Experimental Psychology: Learning Memory and Cognition, 24*, 350-361.

Marsh, R.L., Hicks, J.L., & Bryan, E.S. (1999). The activation of unrelated and canceled intentions. *Memory & Cognition, 27*, 320-327.

Infinite RAAM: A Principled Connectionist Substrate for Cognitive Modeling

Simon Levy and Jordan Pollack
levy, pollack@cs.brandeis.edu
Dynamical and Evolutionary Machine Organization
Volen Center for Complex Systems,
Brandeis University, Waltham, MA 02454, USA
March 1, 2001

Abstract

Unification-based approaches have come to play an important role in both theoretical and applied modeling of cognitive processes, most notably natural language. Attempts to model such processes using neural networks have met with some success, but have faced serious hurdles caused by the limitations of standard connectionist coding schemes. As a contribution to this effort, this paper presents recent work in Infinite RAAM (IRAAM), a new connectionist unification model. Based on a fusion of recurrent neural networks with fractal geometry, IRAAM allows us to understand the behavior of these networks as dynamical systems. Using a logical programming language as our modeling domain, we show how this dynamical-systems approach solves many of the problems faced by earlier connectionist models, supporting unification over arbitrarily large sets of recursive expressions. We conclude that IRAAM can provide a principled connectionist substrate for unification in a variety of cognitive modeling domains.

Language and Connectionism: Three Approaches

Language, to a cognitive scientist, can be held to include natural language and the "language of thought" (Fodor 1975), as well as symbolic programming languages developed to simulate these, like LISP and Prolog. Attempts to build connectionist models of such systems have generally followed one of three approaches.

The first of these, exemplified by (Rumelhart and McClelland 1986), dispenses entirely with traditional representations (data structures) and rules (algorithms on those structures), in favor of letting the network "learn" the patterns in the data being modeled, via the well-known back-propagation algorithm (Rumelhart, Hinton, and Williams 1986) or a similar training method. This approach became the subject harsh criticism from members of the traditional "symbols-and-rules" school of cognitive science, based on the disparity between the strength of the claims made and the actual results reported (Pinker and Prince 1988), as well as the apparent inability of such systems to handle the systematic, compositional aspects of linguistic meaning (Fodor and Pylyshyn 1988).

The second sort of connectionist approach goes beyond the rules-and-representations view and directly to the heart of what computing actually means, by showing how a recurrent neural network can perform all the operations of a Turing machine, or more (Siegelmann 1995). Though such proofs may hold a good deal of theoretical interest, they do not address the degree to which a particular computational paradigm (connectionism) is suited to a particular real-world task (language). They are therefore not of much use in arguing for or against the merits of connectionism as a model of any particular domain of interest (Melnik 2000), any more than knowing about Turing equivalence will help you in choosing between a Macintosh and a Pentium-based PC.

The third approach, which some of its proponents have described as "Representations without Rules" (Horgan and Tienson 1989), is the one that we wish to take here. This approach acknowledges the need for systematic, compositional structure, but rejects traditional, exceptionless linguistic rules in favor of the flexible computation afforded by connectionist representations. Proponents of such a view are of course responsible for showing how these representations can support the kinds of processes traditionally viewed as rules. In the remainder of this paper we show how the behavior of neural network called an Infinite RAAM corresponds directly to one such process, unification, thereby supporting a systematic, compositional model of linguistic structure.

Unification

Unification, an algorithm popularized by Robinson (1965) as a basis for automated theorem-proving, has come to play a central role in both computer science and cognitive science. In computer science, unification is at the core of logical programming languages like Prolog (Clocksin and Mellish 1994); in cognitive science, it is the foundation of a number of category-based approaches to the analysis of natural language (Shieber 1986). The basic unification algorithm can be found in many introductory AI textbooks (e.g., Rich and Knight 1991 p. 152), and can be summarized recursively as follows: (1) A variable can be unified with a literal. (2) Two literals can be unified if their initial predicate symbols are the same and their arguments can be unified.

If, for example, we have a Prolog database containing the assertion `male(albert)`,[1] meaning "Albert is male", and we perform the query `male(Who)`, asking "Who is male?" the unification algorithm will first attempt to unify `male(albert)` with `male(Who)`, and will succeed in matching on the predicate symbol `male`, by rule (2). The algorithm will then recur, attempting to unify the variable `Who` with the atomic literal `albert`, and will succeed by rule (1) and terminate, with the result that `Who` will be bound to `albert`, answering the query.

[1] Prolog examples are taken from the tutorial introduction in (Clocksin and Mellish 1994).