Experiences in Interruptible Language Processing *†

Greg Nelson (ghn@cs.cmu.edu)
School of Computer Science

Jill Fain Lehman (jef@cs.cmu.edu)
School of Computer Science

Bonnie E. John (bej@cs.cmu.edu)
Schools of Computer Science and Psychology
Carnegie Mellon University
Pittsburgh, PA 15213-3891

Appears in Proceedings of the 1994 AAAI Spring Symposium on Active NLP

The Soar NASA Test Director project (NTD-Soar) has sought to develop an integrated agent to be used in evaluating changes to the support technology at the NASA Launch Control Center. The individual being modeled, the NTD, is responsible for coordinating many facets of the testing and preparation that the Space Shuttle must go through before it can be launched. Specifically, he must try to satisfy a checklist of launch procedures that, in its current form, consists of 3000 pages of looseleaf manuals (the Operations and Maintenance Instructions (OMI)), as well as graphical timetables describing the critical timing of particular parts of the launch. To accomplish this, the NTD talks extensively with numerous other members of the launch team over a two-way radio net called the Operational Intercommunications System (OIS). In addition to maintaining a good understanding of the status of the entire launch, the NTD is responsible for coordinating troubleshooting attempts by determining which members of the launch team have the necessary expertise and managing their communication. The goal of simulating the NTD's performance is to develop a method for evaluating proposed user interfaces without jeopardizing the success or efficiency of an actual launch. At the same time, we hope to provide an understanding of the knowledge needed by the NTD to perform his job, which may in turn help in the training of future NTDs.

The NTD's behavior during visual, linguistic, motor, and other cognitive tasks is thoroughly intertwined. Often it is necessary that his tasks be performed in parallel, or at the very least give this appearance. Fodor [3] argues that such behavior can be modeled only by the action of interconnected cognitive modules which do work in parallel. Others have focused on achieving parallelism by finding fixed ways to combine multiple parallel actions into a single larger action (e.g. [2]). Our approach, however, is to explain the apparent parallelism by an ability to switch fluidly and opportunistically among tasks, rather than by actual parallel execution.

During the development of the NTD-Soar model, we have found it useful to distinguish the following concepts in making design decisions:

Interleaving: the method by which we achieve the appearance of doing several things in parallel, despite the requirement that they share a single resource.

Grain size: the size of the smallest unit of processing that can be performed. Alternatively, a commitment to the minimum amount of work that constitutes making progress in the task.

*This research was supported in part by the NASA, Grant No. NAG 2-634 and in part by the Wright Laboratory, Aeronautical Systems Center, Air Force Materiel Command, USAF, and the Advanced Research Projects Agency under grant number F33615-93-1-1330.

†The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of NASA, Wright Laboratory, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

Reactivity: the demand that the rate of progress be in step with the rate of related events in the external environment.

Learning: a method for modifying the performance of the system through the automatic acquisition of new knowledge.

Our purpose here is to briefly outline a number of interrelationships among these concepts that we believe are of general concern to the active NLP community.

The Model

NTD-Soar is an implementation of our model within the Soar cognitive architecture [5]. The model was developed by integrating the existing work on natural language comprehension in Soar [6, 7] with an independently developed model of the page scanning behavior of the NTD [4], and preliminary work on language generation. Although the current model covers only a small subset of the NTD's skills, we have, nevertheless, learned a great deal about the interactions between language and other parts of the task.

Figure 1 shows how the linguistic knowledge in the system is organized into *problem spaces*. Each problem space (triangle) contains *operators* (thin solid arrows) which modify states (circles). Within a Soar model, the operators are the smallest units of deliberate action, and must be performed sequentially. At any point, the knowledge required to unequivocally select or perform an action may be unavailable; in this case, an *impasse* arises (question marks), and a subgoal is created (thin dotted arrows) to search for the knowledge in another problem space. When the subgoal is achieved, a generalization of the problem solving, or *chunk*, is learned, so that in similar future situations, the knowledge will be directly available in the problem space which gave rise to the subgoal. In this way, knowledge recombines and moves up (wide dotted arrows) to be available under all the different circumstances that the agent's experience has shown it to be applicable.

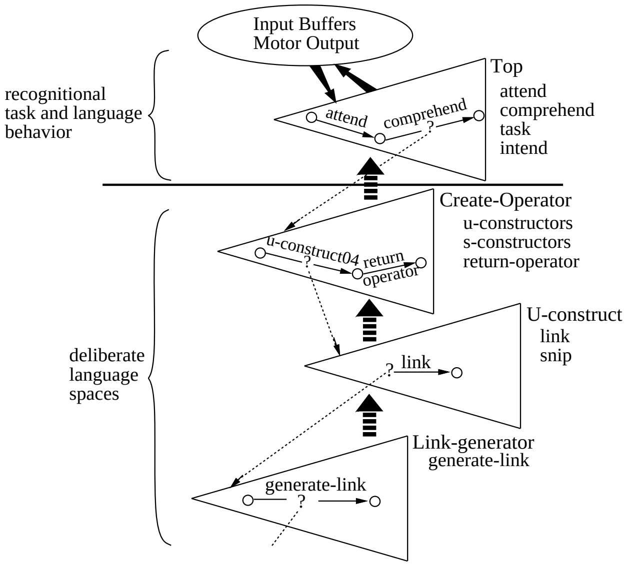

Figure 1: A subset of NTD-Soar's problem spaces.

Figure 1 also distinguishes between recognitional language (and task) behavior as it occurs in the Top space, and deliberate language processing. Recognitional behavior occurs without impasse; it arises through learning over the knowledge sources in the deliberate spaces. Since the real NTD is an expert at his task, we expect his behavior to be largely recognitional.[1] Thus, we consider the system's grain size to be a function of how much work a Top space operator does, i.e., how much integration of knowledge and action is chunked up into the Top space operators. In NTD-Soar, we have found it useful to divide the Top space operators into a number of different categories that reflect the types of basic activities the NTD performs:

Attention operators are used to select for further processing particular elements from a complex representation of the sensory input.

Comprehension operators interpret and elaborate the perceptual inputs in conceptual terms, or perform further inferences on the conceptual representation.

[1] One oddity of the NTD model is that, while we expect most of the real NTD's knowledge to be recognitional, our model starts off with no chunks. This is not a theoretical statement, but a pragmatic decision with significant practical implications.

Task operators select the next task to be performed, thereby biasing the way other operators are selected.

Intend operators initiate physical responses from the system, such as language or simulated hand movements.

All tasks, language included, are performed by collections of these types of operators. Naturally, language comprehension is a form of comprehension operator, and language generation leads to intention operators. However, processing of the visual page is also performed through similar operators – intention operators produce eye movements, and comprehension operators perform the visual pattern matching that allows the NTD to determine whether he has found the information he is looking for. This visual task is quite different from reading, because comprehension is processing neither syntax nor meaning.

In the remainder of this paper, we examine issues arising from the four key concepts introduced above in terms of the model we have described.

Interleaving

Other members of the launch team may summon the NTD at any time, interrupting his otherwise constant flow of visual and motor activity. Words comes into the system's input buffer incrementally and asynchronously with respect to cognition. We model only a limited amount of buffering for auditory input [1]; therefore it is critical that non-linguistic tasks cede control of central cognition on a fairly frequent basis, or the unprocessed input will be lost. For example, since flipping a page requires more than a second to complete [4], this process cannot monopolize the central processor for its entire duration. If it did, the NTD would be unable to comprehend, speak, or make eye movements during this time. In general, the sooner comprehension is allowed to take control, the more time it will have available should comprehension prove to be difficult.

On the other hand, it is equally unreasonable for language comprehension to hold control indefinitely – other inputs, especially if they are critical, must be able to distract the NTD from language processing. Thus, comprehension must proceed *incrementally*, so that there are frequent opportunities for interleaving language with other tasks. Indeed, we can consider the interruption of language and the interruption of non-linguistic tasks to be the same phenomenon. Further, if the comprehension of prior language input is viewed as one task, we can consider the need to process new auditory input to be a distinct task as well, and one equally capable of interrupting the currently selected task. In this light, all combinations of linguistic and non-linguistic tasks as both the current process and the interruption are possible.

One of the keys to NTD-Soar's performance is the presence of specific knowledge that guides the interruption and interleaving of task with language, language with task, and each with itself. There are certainly circumstances in which people simply do not process what others are saying, just as there are situations where a single word is enough to catch one's attention and hold it. To explain these different behaviors, it is necessary to specify the conditions for ignoring potential language tasks, the conditions for interrupting non-language tasks, the conditions for interrupting language tasks, etc. In the absence of data to the contrary, we have made the assumption for this domain that the time to process a single word is sufficiently short that non-linguistic tasks need not abort language processing within a single comprehension operator. Once language is in control, the task always allows language to complete at least one operator's worth of processing. In contrast, because the arrival of new words happens asynchronously and is the impetus for new processing, language *may* interrupt itself.

Grain Size and Reactivity

How much work should a Top space operator do? In other words, what grain size is appropriate to achieve interleaved behavior? It is clearly implausible that language comprehension happens at scales as large as a sentence at a time, but even if we consider the contents of the input buffer to be incremented by lexemic (or phonemic) units, is there one Top operator per lexeme involved in comprehension, or many? And if many, how many?

In the mapping of the Soar architecture onto human cognition, there is an assumption that the selection and application of an operator corresponds to ~50-100 milliseconds of cognitive time [8]. Since recognitional comprehension in humans proceeds at about 250 to 350 milliseconds per word, we have a clear constraint bounding the number of Top space operators per word (similarly, there is a constant for spoken language generation of about 330 to 400 milliseconds per word).

Although implementation in Soar provides this constraint a priori, we can, nevertheless, make some more general observations about the grain size of our operators independent of the mapping to cognition (but not independent of the learning aspect of the architecture). Since language comprehension may be interrupted at any point, either by another task or by the arrival of new input, the amount of work that gets done before the interruption is the critical factor. The more processing that must occur in the subgoal to produce one recognitional comprehension operator, the less often the system will be able to complete that processing, and thus the more slowly language will become recognitional. In the extreme, if the grain size is *too* large, the system will never be able to acquire recognitional language performance as long as the verbal input arrives at normal rates. This limitation can have dramatic effects, including essentially removing the knowledge of a word from the system whenever the structures associated with that word cannot be constructed in the time available.

Of course, we can reduce grain size by reducing the amount of processing each operator does. This means that the system does less processing in each subgoal at the cost of making the operators less powerful. Thus, even after these smaller operators become recognitional, a greater number of them is required to do the same work, leading to a longer total processing time in the Top space. Now instead of racing against time in order to learn something, we are simply pressed to make progress so that we do not fall behind in processing the newly arrived inputs.

The interaction of these two constraints on grain size, with real time performance pushing both for fewer Top space operators (more processing in the subgoals below the Top space) and for smaller, easier to build operators (less processing in the subgoals), greatly narrows the space of workable models. With a running model implemented in Soar, we have been able to test our particular choice of grain size to determine how well it performs. Since the operators are selected serially, we compute the total processing time for a sequence of actions by evaluating the expected duration of each operator in the sequence. Using Soar's prediction of how long each operator takes (~50 milliseconds for highly practiced operators), we can predict the time course of various behaviors and compare this prediction to the real time data at our disposal. With about three recognitional comprehension operators per word, it appears that the model is able to keep up with the flow of input that the NTD receives. These empirical evaluations show that the NL-Soar model *does* succeed in learning recognitional knowledge for sentences presented in real time, albeit in a somewhat surprising manner.

Learning

Though the model is often unable to learn recognitional comprehension operators for the whole input the first time it hears it, it may learn to process some fraction of the words completely. At the very least, because learning happens in each subgoal, the system may learn how to perform more quickly a part of the comprehension of a new word, even if it never completes processing of that word. Since any recognitional processing speeds up comprehension, the *next* time the model hears a related utterance, it processes some parts faster, leaving itself more time to process those parts that it didn't learn the first time through. This behavior can be repeated, with the tendency for some improvement from each utterance the model processes. The result is that the system becomes progressively better at processing language, although it may require multiple trials to learn to process any particular word in a fully recognitional manner.

To see how these types of learning play out in practice, we have evaluated two different types of language comprehension operators in our model. One kind of operator is abruptly terminated (even if nothing has been learned) any time a new word arrives. The other operator is only terminated by the successful completion of the subgoal underneath the comprehension operator (the Create-Operator problem space of Figure 1). These two operators make different predictions. The first says that only partial results will be learned if words are arriving too fast, possibly requiring multiple trials to fully learn the words, but providing a chance to partly process each word that arrives. The second option means that we will always completely learn each word we process, but by the time we are done, the input may have changed sufficiently that the newly learned operator cannot apply. Further, we expect that in some cases, the second strategy will cause us to fall hopelessly behind in real time performance.

Of course, nothing requires us to make this decision once and for all. In fact, in theory we can even make an initial decision and change it as a result of the processing that language comprehension performs. If for some reason we expect the next word to be more important than the current one, we can decide to become interruptible, so that we will have the maximum time available for processing the important word.

Conclusion

We have described a model of language comprehension that can be integrated with other tasks to give the appearance of parallel performance by interleaving task and language. The model operates with a grain size large enough to process input recognitionally in real time, but small enough to ensure that some learning will occur even if the words are coming in too fast for full comprehension by deliberate processing. Although further work needs to be done to determine how well this language model fits with other independently constructed task models, we believe we have made significant progress toward a general understanding of some of the important issues and interactions involved in active NLP.

References

[1] Baddeley, A. *The Concept of Working Memory: A View of its Current State and Probable Future Development.* **Cognition**, Vol. 10 (1981), pp. 17–23.

[2] Covrigaru, A. *Emergence of Meta-Level Control in Multi-Tasking Autonomous Agests.* PhD Thesis, University of Michigan, 1992.

[3] Fodor, J. A. **The Modularity of Mind**. Cambridge, MA: MIT Press, 1983.

[4] John, B. E., Remington, R. W., and Steier, D. M. *An Analysis of Space Shuttle Countdown Activities: Preliminaries to a Computational Model of the NASA Test Director.* Tech Report No. CMU-CS-91-138, Carnegie Mellon University Computer Science, 1991.

[5] Laird, J. E., Newell, A., and Rosenbloom, P. *Soar: An Architecture for General Intelligence.* **Artificial Intelligence**, Vol. 33 (1987), pp. 1–64.

[6] Lehman, J. F., Lewis, R. L., and Newell, A. *Natural Language Comprehension in Soar: Spring 1991.* Tech Report No. CMU-CS-91-117, Carnegie Mellon University Computer Science, 1991.

[7] Lewis, R. L. *An Architecturally-based Theory of Human Sentence Comprehension.* PhD Thesis, Carnegie Mellon University, December 1993. *Also available as Technical Report CMU-CS-93-226.*

[8] Newell, A. **Unified Theories of Cognition**. Cambridge, MA: Harvard University Press, 1990.