Parallel Distributed Processing and Role Assignment Constraints

James L. McClelland
Carnegie-Mellon University

My work in natural language processing is based on the premise that it is not in general possible to recover the underlying representations of sentences without considering semantic constraints on their possible case structures. It seems clear that people use these constraints to do several things:

To assign constituents to the proper case roles and attach then to the proper other constituents.
To assign the appropriate reading to a word or larger constituent when it occurs in context.
To assign default values to missing constituents.
To instantiate the concepts referenced by the words in a sentence so that they fit the context.

I believe that parallel-distributed processing models (i.e., connectionist models which make use of distributed representations) provide the mechanisms that are needed for these tasks. Argument attachments and role assignments seem to require a consideration of the relative merits of competing possibilities (Marcus, 1980; Bates and MacWhinney, 1987; MacWhinney, 1987), as does lexical disambiguation. Connectionist models provide a very natural substrate for these kinds of competition processes (Cottrell, 1985; Waltz and Pollack, 1985).

The use of distributed representations also seems well suited to capturing many aspects of the way people exploit semantic constraints. For choosing between two distinct alternative interpretations of a constituent, local and distributed representations may be approximately equivalent, but distributed representations are much more natural from capturing contextual shading of the interpretation of a constituent. In a distributed representation the pattern of activation that is most typically activated by a particular word or phrase can be subtly shaded by constraints imposed by context; there is no need to limit the choice of alternative shadings to a pre-specified set of alternatives each represented by a differnt single unit. Similarly, filling in missing arguments is not a matter of choosing a particular concept, but of filling in a pattern that specifies what is known about the filler, without necessarily specifying a particular specific concept.

In previous work, Alan Kawamoto and I (McClelland and Kawamoto, 1986) implemented a parallel-distributed processing (PDP) model that can use semantic constraints to do the four things listed at the beginning of the article, though it was limited to processing only one clause at a time. While it would be possible to use such a mechanism clause-by-clause, semantic constraints are often required to decide which of several clauses a phrase belongs to. For example, in the sentence:

1) John ate the cake that his mother baked at the picnic.

we attach "at the picnic" to the main clause (as the place where the cake was eaten), whereas in

2) John ate the cake that his mother baked in the oven.

we attach "in the oven" to the subordinate clause (as the place where the cake was baked). Clearly these attachments depend on knowing that baking can take place in ovens, not at picnics, and eating can take place at picnics, not in ovens; I would also claim that the relative merits of both attachments must be taken into account to get the attachments right. It seems, then, that a mechanism is needed that can consider the possibility of attaching a phrase to more than one possible clause.

This article sketches out a model that aims to achieve multi-clause capability. The model has not yet been fully implemented, so the paper is quite speculative. However, I think the model promises to take us some distance toward a better understanding of the interaction of syntactic and case-role analysis. In particular, it suggests that with the right connectionist architecture, the four uses of semantic constraints enumerated above become intrinsic characteristics of the language processing machinery.

I would like to thank Geoff Hinton, George Lakoff, Brian MacWhinney and Mark St. John for discussions of the topic of this paper and/or for specific comments on the first draft. Supported by ONR contract N00014-82-C-0374, NR 667-483.

Representing structure and content. To begin, let us consider how to represent the structure of a sentence in a PDP mechanism. To do this, we make use of the notion that a structural description can be repesented as a set of triples. For example the correct role structure of Sentence 2 can be represented with a set of triples such as the following:

(P1 AGENT BOY) (P1 ACTION ATE) (P1 PATIENT CAKE)
(P2 AGENT MOTHER) (P2 ACTION BAKED) (P2 PATIENT CAKE)
(P2 LOCATION OVEN)

An individual triple can be represented in distributed form by dedicating a set of units to each of its parts; thus we can have one set of units for the head of the triple, one for the relation, and one for the tail or slot-filler. Each of the three parts of a triple can then be represented in distributed form as a pattern of activation over the units. The idea of using this kind of three-part distributed representation was introduced by Hinton (1981) to represent the contents of semantic nets; the extension to arbitrary tree structures is due to Touretzky and Hinton (1985) and Touretzky (1986).

For the fillers, or the tail of a triple, the units stand for useful characterizers that serve to distinguish one filler from another. Hinton (1981) used the term "microfeatures" for these units; these features need not correspond in any simple way to verbalizable primitives. Different slot fillers produce different patterns on these units; and the different possible instantiations of a filler are likewise captured by differences in the pattern of activation on the units.

For the relations, the units stand for characteristics of the relation itself. Note that this differs from most other approaches in treating each role or relation as a distributed pattern. This has several virtues. For one thing, it immediately eliminates the problem of specifying a small set of case roles, in the face of the fact that there seem to be a very large number of very subtle differences between roles that are in many ways very similar. Further, the use of distributed representations allows us to capture both the similarities and differences among case roles. The idea has been proposed on independent linguistic grounds, as well.

For the head of each triple, the units stand for characteristics of the whole in which the filler plays a part. Thus the pattern that represents P1 is not some arbitrary pointer as it might be in a Lisp-based representation, but is rather a *Reduced Description* of the constituent that it stands for (Hinton, McClelland, and Rumelhart, 1986; Lakoff, personal communication). In particular, the pattern representing P1 would capture characteristics of the act of eating and of the participants in the act. There would be less detail, of course, than in the separate representations of these constituents where they occur as separate fillers of the tail slot.

Syntactic and case-role representations. Sentences have both an augmented surface structure representation and a case-role representation. In the present model, then, there are two sets of units, one that represents the syntactic structure triples, and one that represents the case-structure triples. I have already described the general form of the case-role triples; the syntactic triples would have a similar form, though they would capture primarily syntactic relations among the constituents. So, for example, the set of syntactic triples of Sentence 2 would be something like:

(S1 SUBJ BOY) (S1 VERB SAW) (S1 DOBJ CAKE)
(CAKE MODIFIER S2)
(S2 SUBJ MOTHER) (S2 VERB BAKED) (S2 DOBJ T=CAKE)
(S1 LOC-PP OVEN)

There are, correspondingly, two main parts to the model, a syntactic processor and a case-frame processor (See Figure 1). In this respect, the model is similar to many conventional parsing schemes (e.g., Marcus, 1980; Kaplan and Bresnan, 1982). The microstructure is quite different, however. One of the key things that a PDP microstructure buys us is the ability to improve the interaction between these two main components.

Syntactic processing. The role of the syntactic processor is to take in words as they are encountered in reading or listening and to produce at its outputs a sequence of patterns, with each pattern capturing one syntactic structure triple.[1] In Figure 1 the syntactic processor is shown in the midst of processing Sentence 2. It has reached the

1. Note that this means that several words can be packed into the same constituent, and that as the words of a constituent (e.g., "the old grey donkey") are encountered the microfeatures of the constituent will be gradually specified. Thus the representation of the constituent can gradually build up at the output of the syntactic processor.

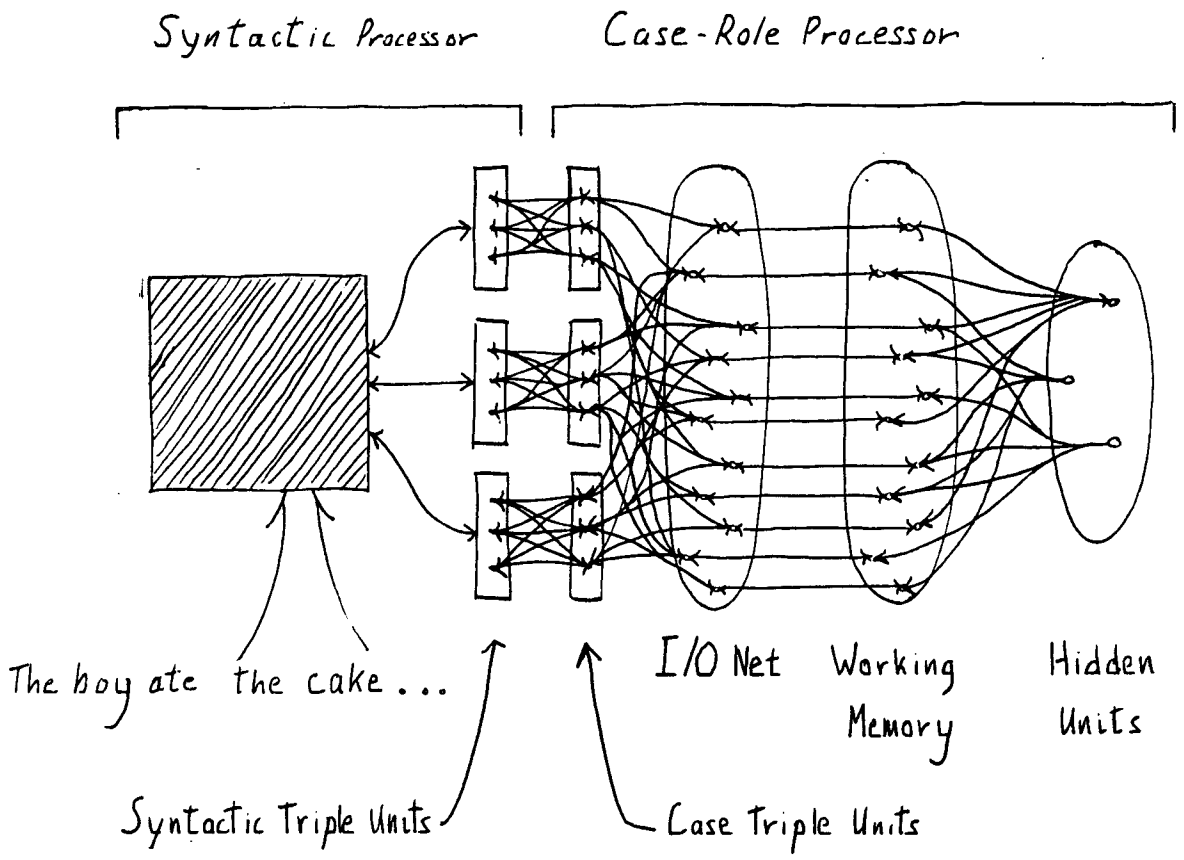

Figure 1. A diagram of the model. See text for explanation.

point where it is processing the words "the cake". The output of at this point should tend to activate the pattern corresponding to (S1 DOBJ CAKE) over a set of units (the *syntactic triple units*) whose role is to display the pattern of activation corresponding to the current syntactic triple. Note that these units also receive feedback from the case-frame processor; the role of this feedback is to fill in unspecified parts of the syntactic triple, as shall be discussed below. The syntactic triple units have connections to units (the *case-frame triple units*) which serve to represent the current case-frame triple.

The connections between these two sets of units are assumed to be learned through prior pairings of syntactic triples and case-frame triples, so that they capture the mutual constraints on case and syntactic role assignments. The inner workings of the syntactic processor have yet to be fully worked out, so for now I leave it as a black box.

The case-frame processor. The role of the case-frame processor is to produce an active representation of the current case-frame constituent, based on the pattern representing the current syntactic constituent on the syntactic triple units and on feedback from a set of units called the *working memory*. The working memory is the structure in which the developing case-frame representation of the sentence is held. As constituents are parsed, they are loaded into the working memory, by way of a network called an I/O net.[2] Within the working memory, individual units correspond to combinations of units in the current case-role representation. Thus, the representation at this level is *conjunctive*, and is therefore capable of maintaining information about which combinations of case-role units were activated together in the same case-role triple when the patterns activated by several triples are superimposed in the working memory (see Hinton et al 1986, for discussion). Of course, early in a parse, the loaded constituents will necessarily be incomplete.

Pattern completion. The working memory provides a persisting representation of the constituents already parsed. This representation persists as a pattern of activation, so that it can both constrain and be constrained by new constituents as they are encountered, through interactions with a final set of units, called the *hidden case-role units*. These units are called "hidden" because their state is not visible to any other part of the system; instead they

2. The I/O net is equivalent to Touretzky and Hinton's (1985) "pull-out net". Its job is to ensure that the characteristics of only a one of the constituents stored in the working memory are interacting with the case-frame triple units. See Touretzky and Hinton (1985) for details.

serve to mediate constraining relations among the units in the working memory. The process works as follows. Connections from working memory units to hidden units allow the pattern of activation over the working memory to produce a pattern over the hidden units. Connections from the hidden units to the working memory units allow these patterns, in turn, to feed activation back to the working memory. This feedback allows the network to complete and clean-up distorted and incomplete patterns (that is, representations of sentences). The connections in the network are acquired through training on on a sample of sentences (see St. John, 1986, for details). The connection strengths derived from this training experience allow it to sustain and complete the representations of familiar sentences; this capability generalizes to novel sentences with similar structure.

What this model can do. The model I have described should be able to do all of the kinds of things listed at the beginning of the paper. Consider, for example, the problem of interpreting the sentence "The boy hit the ball with the bat." This requires both assigning the appropriate reading (baseball bat) and the appropriate role (instrument) to the bat. The syntactic triple for this constituent (S1 with-PP BAT), would tend to activate a pattern over the coresponding to a blend of baseball bat and flying bat as the tail of the triple, and a blend of the possible case-roles consistent with "with" as the the pattern representing the relation portion of the triple. These in turn would tend to activate units representing the various possible filler-role combinations consistent with this syntactic constituent. But since the other constituents of the sentence would already have been stored in the working memory, the completion process would tend to support units standing for the baseball-bat as instrument interpretation more than others. Thus, simultaneous role assignment and context sensitive selection of the appropriate reading of an ambiguous word would be expected to fall out naturally from the operation of the completion process.

Filling in default values for missing arguments and shading or shaping the representations of vaguely described constituents is also a simple by-product of the pattern completion process. Thus, for example, on encountering "The man stirred the coffee", the completion process will tend to fill in the pattern for the completion that includes a spoon as instrument. Note that the pattern so filled in need not specify a particular specific concept; thus for a sentence like "The boy wrote his name", we would expect a pattern representing a writing instrument, but not specifying if it is a pen or a pencil, to be filled in; unless, of course, the network had had specific experience indicating that boys always write their names with one particular instrument or another. A similar process occurs on encountering the container in a sentence like "The container held the cola". In such cases the constraints imposed by other constituents (the cola) would be expected to shape the representation of "container", toward a smallish, hand-holdable, non-porous container; Again, this process would not necessarily specify a specific container, just the properties such a container could be predicted to have.

I have not yet said anything about what the model would do with the attachment problem posed by the sentence "The boy ate the cake that his mother baked in the oven." In this case, we would expect that the syntactic processor would pass along a constituent like (S? in-PP OVEN), and that it would be the job of the case-role processor to determine its correct attachment. Supposing that the experience the network has been exposed to includes mothers (and others) baking cakes (and other things) in ovens, we would expect that the case-role triple (P2 LOC OVEN) (where P2 stands for the reduced description of "mother-baked-cake") would already be partially active as the syntactic constituent became available. Thus the incoming constituent would simply reinforce a pattern of activation that already reflected the correct attachment of oven.

Current status of the model. As I previously stated, the model has not yet been implemented, and so one can treat the previous section as describing the performance of a machine made out of hopeware. Nevertheless I have reason to believe it will work. CMU connectionists now have considerable experience with representations of the kind used in the case-frame processor (Touretzky & Hinton, 1985; Touretzky, 1986; Derthick, 1986). A mechanism quite like the case-frame processor has been implemented by St. John (1986), and it demonstrates several of the uses of semantic constraints that I have been discussing.

Obviously, though, even if the case-frame processor is successful there are many more tasks that lie ahead. One crucial one is the development of a connectionist implementation of the syntactic processor. I believe that we are now on the verge of understanding sequential processes in connectionist networks (see Jordan, 1986), and that this will soon make it possible to describe a complete connectionist mechanism for language processing that captures both the strengths and limitations of human language processing capabilities.

References

Bates, E., & MacWhinney, B. (1987). Competition, variation and language learning: What is not universal in language acquisition. In B. MacWhinney (Ed.), *Mechanisms of language acquisition.* Hillsdale, NJ: Erlbaum.

Cottrell, G. (1985). *A connectionist approach to word sense disambiguation* (TR-154). Rochester, NY: University of Rochester, Department of Computer Science.

Derthick, M. (1986). *A connectionist knowledge representation system.* Thesis proposal, Carnegie-Mellon University, Department of Computer Science, Pittsburgh, PA.

Hinton, G. E. (1981). Implementing semantic networks in parallel hardware. In G. E. Hinton & J. A. Anderson (Eds.), *Parallel models of associative memory* (pp. 161-188). Hillsdale, NJ: Erlbaum.

Hinton, G. E., McClelland, J. L., & Rumelhart, D. E. (1986). Distributed Representations. In D. E. Rumelhart, J. L. McClelland, & the PDP research group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition. Volume I.* Cambridge, MA: Bradford Books.

Jordan, M. I. (1986). *Serial order: A parallel distributed processing approach* (ICS Rep. No. 8604). University of California, San Diego, Institute for Cognitive Science.

Kaplan, R., & Bresnan, J. (1982). Lexical functional grammar: A formal system for grammatical representation. In J. Bresnan (Ed.), *The mental representation of grammatical relations.* Cambridge, MA: MIT Press.

Kawamoto, A. H. (1985). *Dynamic processes in the (re)solution of lexical ambiguity.* Unpublished doctoral dissertation, Brown University.

MacWhinney, B. J. (1987). The competition model. In B. MacWhinney (Ed.), *Mechanisms of language acquisition.* Hillsdale, NJ: Erlbaum.

Marcus, M. P. (1980). *A theory of syntactic recognition for natural language.* Cambridge, MA: MIT Press.

McClelland, J. L. (in press.) How we use what we know in reading: An interactive activation approach. In M. Coltheart (Ed.), *Attention and performance XII: The psychology of reading.* London: Erlbaum.

McClelland, J. L., & Kawamoto, A. H. (1986). Mechanisms of sentence processing: Assigning roles to constituents. In J. L. McClelland, D. E. Rumelhart, & the PDP research group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition. Volume II.* Cambridge, MA: Bradford Books.

St. John, M. F. (1986). *Reconstructive memory for sentences.* Working paper, Department of Psychology, Carnegie-Mellon University, Pittsburgh, PA.

Touretzky, D. S. (1986). BoltzCONS: Reconciling connectionism with the recursive nature of stacks and trees. *Proceedings of the Eighth Annual Conference of the Cognitive Science Society,* Amherst, MA, 522-530.

Touretzky, D., & Hinton, G. E. (1985). Symbols among the neurons: Details of a connectionist inference architecture. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence.*

Waltz, D. L., & Pollack, J. B. (1985). Massively parallel parsing. *Cognitive Science, 9,* 51-74.