REDUCING SEARCH BY PARTITIONING THE WORD NETWORK

John Dowding
Unisys Paoli Research Center
P.O. Box 517
Paoli, PA 19355
dowding@prc.unisys.com

This paper proposes an architecture for integrating speech recognition and natural language processing to provide a spoken language understanding system. This work, done in collaboration with the MIT Spoken Language Systems Group, has enabled us to interface our Pundit natural language processing system [Dahl1987,Palmer1986,Hirschman1982,Hirschman1986,Dowding1987] and SUMMIT, the MIT speech recognition system[Zue1989,Glass1988,Seneff1985,Zue1985]. Information is passed between the two systems using the *Word Network* (or *Word Lattice*) which is a set of word-score pairs together with the start and end points for each word. The word network is organized as a directed acyclic graph, whose arcs are labeled as word-score pairs, and whose nodes are moments in time. The recognition problem is to find the best scoring grammatical sequence of words from the graph's begin-point to its end. In our experments, we analyzed word networks from the TIMIT domain and the Resource Management domain, constructed without using any language model.

A spoken language system requires the coupling of speech recognition capabilities with a capability for understanding the meaning of the utterance as provided by a natural language understanding module. The architecture of such a system is an active topic of research. One approach is to simply couple the two components sequentially: the speech recognizer converts the speech signal into a sequence of words, and then the natural language system attempts to understand those words. The criticism of this approach is that natural language system has no opportunity to correct the errors made by the recognizer; nor can it help in controlling the search space. A variant to the this approach is to have the speech recognizer produce a ranked set of candidate sentences. These sentences are then be processed by the natural language system, and the best scoring grammatical (or meaningful) sentence accepted. Alternatively, the two systems can be integrated at a much deeper level, allowing natural language constraints to provide assistance to the recognizer by scoring sentence prefixes as they are produced by the recognizer; this would have the effect of using the grammar to prune the search space, while also producing a representation of the meaning of the utterance. This latter approach -- close interleaving of the speech and natural language components -- is our architectural goal. However, in order to test the effectiveness of these ideas, we have experimented with a sequential coupling of the two systems where the speech recognizer produces a word network, from which candidate sentences are extracted and processed by the natural language component.

Our initial problem in coupling the MIT speech recognition system with the Unisys PUNDIT natural language system was to define an appropriate interface for experimentation. The MIT speech recognizer uses a method called *Viterbi Search* to find the highest scoring sentence. This search strategy is able to find the highest scoring sentence, but it can not be used in its current form to find successively lower scoring sentences. If we had adhered to this strategy, the natural language system might have accepted or rejected the unique output of the speech recognition component, but, as described above, could not have participated in defining a more intelligent search.

In order to allow the natural language system to examine more than one top-scoring sentence candidate, MIT modified their speech recognizer to produce the word network. By developing a strategy to

generate candidate sentences from the word network, it is now possible to couple PUNDIT to this system as a filter. The recognizer generates the word network, a search procedure traverses this network, producing candidate sentences (or sentence prefixes) and PUNDIT accepts or rejects these sentences (or sentence prefixes). Thus if the top scoring string of words turns out not to be a meaningful sentence, there may still be an acoustically lower scoring candidate that is meaningful and receives an analysis from PUNDIT.

Long term, this architecture will permit us to couple PUNDIT to the word network search, to provide its filtering as the word string is built up. These results, however, report on the use of PUNDIT to filter entire candidate sentences. We have done some experiments on the use of PUNDIT to prune sentence prefixes, and have found that this tends to underestimate the power of natural language constraints, since these constraints are much stronger at the end of the sentence than they are for any of its prefixes.

The main contribution of this paper is to outline the search strategy used to traverse the word network at the interface between the speech recognizer and the natural language component. We experimented with various search strategies to determine the best approach to search the word networks. These strategies included a mix of admissible (breadth-first and best-first) and inadmissible (beam search) strategies. We found that all of the admissible search strategies were too inefficient to be practical. They could not deal with sentence lengths greater than 4 words before they became too slow for our current hardware (Sun 3/60 Workstations). Using an inadmissible search, we could increase efficiency by reducing the beam size, but then we could not get the correct answer. We experienced the phenomenon that the highest scoring path in a beam early in the search would have so many descendents that it would totally dominate the beam later in the search, yielding beams where the alternative candidates were very similar. Because the search was performed strictly left to right, all of the candidates in the beam would share the same left prefix. A similar phenomenon occurs when using an island-driven strategy, except that the common portion may not occur on the left.

We then designed an alternative approach that does not suffer from these problems. This approach maintains a queue of high-scoring candidates that may come from any part of the word network, and that may not be similar to each other. Despite the fact that this approach also uses a strict left to right search, there is no left prefix bias among the high scoring candidates. We partition the word network such that all words that have the same start and end points belong to the same partition. Within each partition, the word-score pairs are sorted by score. The score of a partition is defined to be the score of its highest scoring word. This partitioning dramatically reduces the size of the graph that we are searching. For instance, the graph for the sentence "Barb's gold bracelet was a graduation present." contains 1868 arcs. The resulting graph after partitioning contained only 489 arcs (with the same number of nodes in both graphs). This partitioning reduces the "bushiness" of the graph, allowing traditional search procedures to be effective. Figure 1 contains a part of a sample partition from this word network. For 6 of the 8 partitions in this partition path, the correct word (including the pause) was the top candidate within its partition. Notice however that there is still a significant amount of search remaining due to the difference between the scores for "large" and "barb's" in partition 2, especially when compared to the differences between "a", "and", "if", "in", and "an" in partition 6.

Under this scheme, the search for the correct path through the word network is done in two parts: First, the partitioned graph is searched to find the highest scoring partition paths. This search can be done using either an admissible or a beam search strategy. Second, the high scoring sentences are extracted from the set of high scoring partition paths. The sentences are extracted one at a time in order of highest score until one is found that is acceptable to the natural language components.

The search for the highest scoring partitions can be done very quickly, and the beam size can be very small. The worst-case performance of this search is quite good. The amount of time that it takes to find the highest scoring partitions increases with sentence length and beam size, but is nearly unaffected by vocabulary size (worst case growth is O(N log N) as vocabulary increases, but only due to the need to sort the word-score pairs within each partition). Currently, the score of a partition path is the sum of the score of the highest scoring word in each partition. However this algorithm is independent of the particular scoring algorithm used to combine word scores into word-path scores. We plan on

1		2		3		4		5		6		7		8	
-pau-	-9	large	-58	caught	-67	bracelet	-102	was	-51	a	-23	graduation	-157	present	-138
		i'd	-116	gold	-124	geese	-155	with	-89	and	-23	countryside	-189	pairs	-143
		right	-160	could	-134					if	-23			paper	-147
		like	-165							in	-23				
		guard	-193							an	-86				
		barb's	-188												

Figure 1. Sample Partition for "Barb's gold bracelet was a graduation present"

experimenting with more sophisticated scoring techniques, including density and short-fall scoring[Woods1982], in the future.

Extracting the highest scoring sentences from the high scoring partition paths is also done efficiently. The algorithm to do this is simple: The partition paths are maintained in a priority queue. To find the highest scoring sentence, the top partition path is removed from the queue and its highest scoring sentence is extracted and reported as the highest scoring sentence in the queue. Then the second highest scoring sentence in the partion path is found, and its score becomes the new score for that partition path. It is then returned to the priority queue based on its new score. The loop is repeated as often as necessary until a sentence is found that is acceptable to the natural language components. While the worst-case performance of this part of the search is not good (the amount of time it takes to find the next highest scoring sentence can grow exponentially), the practical performance is much better, and is able to find the top candidates very quickly. Those cases in which the correct sentence gets a very low score will take a long time to find, but that will be true of any search.

We have tested this interface on word networks from both the Timit and Resource Management domains. We computed the word accuracy for 10 Resource Management word networks chosen randomly from the test set. These networks averaged 4600 arcs (drawn from a vocabulary of 991 words) for sentences ranging in length from 4-11 words. The word accuracy figures for these networks are reported in Figure 2. For comparison, the word accuracy of the SUMMIT system on the same 10 sentences is included. These figures deserve some explanation. The word accuracy figure for the no-

Word Networks		
	Word Accuracy	Perplexity
No Grammar	51%	1126
Syntax	64%	1064
Word-Pair Grammar	85%	60
SUMMIT		
No Grammar	61%	991
Word-Pair Grammar	86%	60

Figure 2. Word Accuracy Results

grammar case is computed by comparing the highest scoring candidate to the correct sentence. We then computed the word accuracy for Pundit by having the search procedure generate candidates one at a time until one was found that was acceptable to Pundit. This experiment used only Pundit's syntactic component. We expect better results using Pundit's semantic and pragmatic components and will report on these results at a subsequent meeting. Finally, we computed the word accuracy using our word network traversal procedure but with the word-pair grammar in place of Pundit's syntactic component.

The difference in word accuracy between the no-grammar case for the word networks (51%) and for SUMMIT (61%) is attributable to two causes. First, the word networks used in our experiments are not complete. They were generated by computing the best score for each word ending at each point. A complete network would have to compute the best score for each word at every beginning and end point. The SUMMIT system does not use an explicit word network, but has access to the complete set of begin and end points for all words. Second, our search for the highest scoring candidate used a beam search (beam size = 500) which is not an admissible search. When computing the word accuracy score for the word-pair grammar, only eight of the ten networks produced candidates that were acceptable to the word-pair grammar within the top 5000 candidates. We expect that this behavior will not occur when the grammar checking and word network search are fully interleaved, because the grammar will prune away ungrammatical paths earlier, permitting well-formed candidates to make it into the beam. For comparison, the performance of Pundit's grammar on the same 8 networks was 72%.

Also in Figure 2 are the perplexity results for Pundit's grammar. This number (1064) appears larger than the vocabulary size (991). We have added vocabulary items representing idiomatic expressions to increase the total vocabulary size to 1126. There are several reasons why this perplexity is so high. The grammar used by Pundit is a very broad coverage grammar of English, including sentence fragments and compound sentences. Also, the lexical items were defined keeping their complete English definition in mind. We are developing a methodology for using the training data for a domain to automatically constrain the broad coverage grammar and lexicon. Finally, our grammar of English provides much stronger constraints at the end of a sentence than it does at any other point. Perplexity only captures constraints that provide immediate pruning of a path. If a constraint would eventually block a path, this is not captured in the perplexity computation. Despite the high perplexity of the Pundit grammar, 77% of the sentence candidates produced by the network traversal procedure were rejected by Pundit.

We are very optimistic that this search procedure will scale up to use in a fully intergrated Spoken Language system. In the prototype, pruning of the search space by the natural language components was only done as the very last step in the search. It is possible to implement this search such that the pruning is done incrementally through the search. This will allow us to both reduce the beam size, and increase the probability that the correct answer will be one of the top candidates at the end of the search.

I would like to acknowledge several people who assisted in this work. Michael Phillips of the MIT Spoken Language Systems Group constructed the word networks, and gave very helpful advice on how they should be searched. Lynette Hirschman, Deborah Dahl, and Shirley Steele read an earlier version of this paper, and gave helpful comments. Francois-Michel Lang helped me find an efficient algorithm for extracting high scoring sentences from a partition path.

References

Dahl1987

Deborah A. Dahl, John Dowding, Lynette Hirschman, Francois Lang, Marcia Linebarger, Martha Palmer, Rebecca Passonneau, and Leslie Riley, Integrating Syntax, Semantics, and Discourse: DARPA Natural Language Understanding Program, R&D Status Report, Paoli Research Center, Unisys Defense Systems, May 14, 1987.

Dowding1987

John Dowding and Lynette Hirschman, Dynamic Translation for Rule Pruning in Restriction Grammar, Presented at the 2nd International Workshop on Natural Language Understanding and Logic Programming, Vancouver, B.C., Canada, 1987.

Glass1988

James R. Glass and Victor W. Zue, Multi-Level Acoustic Segmentation of Continuous Speech, Presented at the International Conference on Acoustics, Speech, and Signal Processing, New York, NY, April 11-14, 1988.

Hirschman1982

L. Hirschman and K. Puder, Restriction Grammar in Prolog. In *Proc. of the First International Logic Programming Conference*, M. Van Caneghem (ed.), Association pour la Diffusion et le Developpement de Prolog, Marseilles, 1982, pp. 85-90.

Hirschman1986

L. Hirschman, Conjunction in Meta-Restriction Grammar. *J. of Logic Programming* **3**(4), 1986, pp. 299-328.

Palmer1986

Martha S. Palmer, Deborah A. Dahl, Rebecca J. [Passonneau] Schiffman, Lynette Hirschman, Marcia Linebarger, and John Dowding, Recovering Implicit Information, Presented at the 24th Annual Meeting of the Association for Computational Linguistics, Columbia University, New York, August 1986.

Seneff1985

Stephanie Seneff, Pitch and Spectral Analysis of Speech Based on an Auditory Synchrony Model, PhD Thesis, Massachusetts Institute of Technology, 1985.

Woods1982

W.A. Woods, Optimal Search Strategies for Speech Understanding Control. *Artificial Intelligence* **18**, 1982.

Zue1989

Victor Zue, James Glass, Michael Phillips, and Stephanie Seneff, Acoustic Segmentation and Phonetic Classification in the SUMMIT System, To Be Presented at the International Conference on Acoustics, Speech, and Signal Processing, Glasgow, Scotland, May 23-26, 1989.

Zue1985

Victor W. Zue, The Use of Speech Knowledge in Automatic Speech Recognition. *Proceedings of the IEEE* **73**(11), 1985, pp. 1602-1615.