Lexical Disambiguation using Simulated Annealing

Jim Cowie, Joe Guthrie, Louise Guthrie

Computing Research Laboratory
Box 30001
New Mexico State University
Las Cruces, NM 88003-0001

ABSTRACT

The resolution of lexical ambiguity is important for most natural language processing tasks, and a range of computational techniques have been proposed for its solution. None of these has yet proven effective on a large scale. In this paper, we describe a method for lexical disambiguation of text using the definitions in a machine-readable dictionary together with the technique of simulated annealing. The method operates on complete sentences and attempts to select the optimal combinations of word senses for all the words in the sentence simultaneously. The words in the sentences may be any of the 28,000 headwords in Longman's Dictionary of Contemporary English (LDOCE) and are disambiguated relative to the senses given in LDOCE. Our initial results on a sample set of 50 sentences are comparable to those of other researchers, and the fully automatic method requires no hand coding of lexical entries, or hand tagging of text.

1. Introduction

The problem of word-sense disambiguation is central to text processing. Recently, promising computational methods have been suggested [Lesk, 1987; McDonald et al., 1990; Wilks et al., 1990; Zernik and Jacobs, 1990; Guthrie et al., 1991; Hearst, 1991] which attempt to use the local context of the word to be disambiguated together with information about each of its word senses to solve this problem.

Lesk [1987] described a technique which measured the amount of overlap between a dictionary sense definition and the local context of the word to be disambiguated to successfully disambiguate the word "cone" in the phrases "pine cone" and "ice cream cone". Later researchers have extended this basic idea in various ways. Wilks et al., [1990] identified neighborhoods of the 2,187 control vocabulary words in Longman's Dictionary of Contemporary English (LDOCE) [Procter, 1978] based on the co-occurrence of words in LDOCE dictionary definitions. These neighborhoods were then used to expand the word sense definitions of the word to be disambiguated, and the overlap between the expanded definitions and the local context was used to select the correct sense of a word. A similar method reported by Guthrie et al., [1991] defined subject-specific neighborhoods of words, using the subject area markings in the machine readable version of LDOCE. Hearst [1991] suggests using syntactic information and part-of-speech tagging to aid in the disambiguation. She gathers co-occurrence information from manually sense-tagged text. Zernik and Jacobs [1990] also derive their neighborhoods from a training text which has been sense-tagged by hand. This method incorporates other clues as to the sense of the word in question found in the morphology or by first tagging the text as to part of speech.

Although each of these techniques looks somewhat promising for disambiguation, the techniques have only been applied to several words, and the results have been based on experiments which repeatedly disambiguate a single word (or in [Zernik and Jacobs, 1990], one of three words) in a large number of sentences. In the cases where a success rate for the technique is reported, the results vary from 35% to 80%, depending on whether the correct dictionary sense is desired, or some coarser grained distinction is considered acceptable.

For even the most successful of these techniques, processing of text is limited because of the amount of computation necessary to disambiguate each word in a sentence. A sentence which has ten words, several of which have multiple senses, can easily generate a million possible combinations of senses. The following figure ?? illustrates the number of combinations of word senses in the example sentences used in our experiment described below. Furthermore, if only one sense is computed at a time, as is the case in all of the numerically based work on disambiguation, the question arises as to whether and how to incorporate the fact that a sense has been chosen for a word when attempting to disambiguate the next. Should this first choice be changed in light of how other word senses are selected? These problems have not yet been addressed.

In contrast to the somewhat numerical techniques described above, more principled methods based on linguistic information such as semantic preferences [Wilks, 1975a; 1975b; Wilks and Fass, 1991] have also been used for lexical disambiguation. These methods require exten-

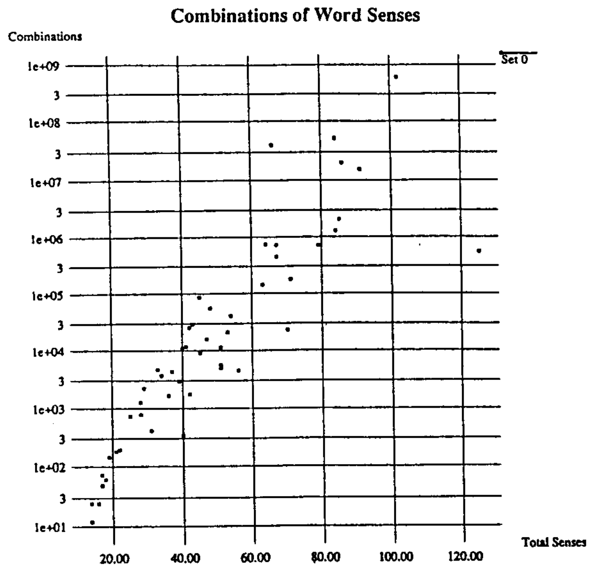


Figure 1:

sive hand crafting by specialists of lexical items: assigning semantic categories to nouns, preferences to verbs and adjectives, etc. Maintaining consistency in these categories and preferences is a problem, and these methods are also susceptible to the combinatorial explosion described above.

In this paper we suggest the application of a computational method called simulated annealing to this general class of methods (including some of the numerical methods referenced above) to allow all senses to be determined at once in a computationally effective way. We describe the application of simulated annealing to a basic method similar to that of Lesk [1987] which doesn't make use of any of the features such as part of speech tagging, subject area, or the use of morphology to determine part of speech. The simplicity of the technique makes it fully automatic, and it requires no hand-tagging of text or hand-crafting of neighborhoods. When this basic method operates under the guidance of the simulated annealing algorithm, sense selections are made concurrently for all ambiguous words in the sentence in a way designed to optimize their choice. The system's performance on a set of test sentences was encouraging and can be expected to improve when some of the refinements mentioned above are incorporated.

2. Simulated Annealing

The method of simulated annealing [Metropolis et al., 1953; Kirkpatrick et al., 1983] is a technique for solving large scale problems of combinatorial minimization. It has been successfully applied to the famous traveling salesman problem of finding the shortest route for a salesman who must visit a number of cities in turn, and is now a standard method for optimizing the placement of circuit elements on large scale integrated circuits. Simulated annealing was applied to parsing by Sampson [1986], but since the method has not yet been widely applied to Computational Linguistics or Natural Language Processing, we describe it briefly.

The name of the algorithm is an analogy to the process by which metals cool and anneal. A feature of this phenomenon is that slow cooling usually allows the metal to reach a uniform composition and a minimum energy state, while fast cooling leads to an amorphous state with higher energy. In simulated annealing, a parameter T which corresponds to temperature is decreased slowly enough to allow the system to find its minimum.

The process requires a function E of configurations of the system which corresponds to the energy. It is E that we seek to minimize. From a starting point, a new configuration is randomly chosen, and a new value of E is computed. If the new E is less than the old one, the new configuration is chosen to replace the older. An essential feature of simulated annealing is that even if the new E is larger than the old (indicating that this configuration is farther away from the desired minimum than the last choice), the new configuration may be chosen. The decision of whether or not to replace the old configuration with the new inferior one is made probabilistically. This feature of allowing the algorithm to "go up hill" helps it to avoid settling on a local minimum which is not the actual minimum. In succeeding trials, it becomes more difficult for configurations which increase E to be chosen, and finally, when the method has retained the same configuration for long enough, that configuration is chosen as the solution. In the traveling salesman example, the configurations are the different paths through the cities, and E is the total length of his trip. The final configuration is an approximation to the shortest path through the cities. The next section describes how the algorithm may be applied to word-sense disambiguation.

3. Word-Sense Disambiguation

Given a sentence with N words, we may represent the senses of the ith word as $s_{i1}, s_{i2}, \ldots s_{ik_i}$, where k_i is the number of senses of the ith word which appear in LDOCE. A configuration of the system is obtained by choosing a sense for each word in the sentence. Our goal is to choose that configuration which a human disambiguator would choose. To that end, we must define a function E whose minimum we may reasonable expect to correspond to the correct choice of the word senses.

The value of E for a given configuration is calculated in terms of the definitions of the N senses which make it up. All words in these definitions are stemmed, and the results stored in a list. The redundancy R is computed by giving a stemmed word form which appears n times a score of $n-1$ and adding up the scores. Finally, E is defined to be $\frac{1}{1+R}$.

The rationale behind this choice of E is that word senses which belong together in a sentence will have more words in common in their definitions (larger values of R) than senses which do not belong together. Minimizing E will maximize R and determine our choice of word senses.

The starting configuration C is chosen to be that in which sense number one of each word is chosen. Since the senses in LDOCE are generally listed with the most frequently used sense first, this is a likely starting point. The value of E is computed for this configuration. The next step is to choose at random a word number i and a sense S_{ij} of that ith word. The configuration C' is is constructed by replacing the old sense of the ith word by the sense S_{ij}. Let ΔE be the change from E to the value computed for C'. If $\Delta E < 0$, then C' replaces C, and we make a new random change in C'. If $\Delta E >= 0$, we change to C' with probability $P = e^{-\frac{\Delta E}{T}}$. In this expression, T is a constant whose initial value is 1, and the decision of whether or not to adopt C' is made by calling a random number generator. If the number generated is less than P, C is replaced by C'. Otherwise, C is retained. This process of generating new configurations and checking to see whether or not to choose them is repeated on the order of 1000 times, T is replaced by $0.9\,T$, and the loop entered again. Once the loop is executed with no change in the configuration, the routine ends, and this final configuration tells which word senses are to be selected.

4. An Experiment

The algorithm described above was used to disambiguate 50 example sentences from LDOCE. A stop list of very common words such as "the", "as", and "of" was removed from each sentence. The sentences then contained from two to fifteen words, with an average of 5.5 ambiguous words per sentence. Definitions in LDOCE are broken down first into broad senses which we call "homographs", and then into individual senses which distinguish among the various meanings. For example, one homograph of "bank" means roughly "something piled up." There are five senses in this homograph which distinguish whether the thing piled up is snow, clouds, earth by a river, etc.

Results of the algorithm were evaluated by having a literate human disambiguate the sentences and comparing these choices of word senses with the output of the program. Using the human choices as the standard, the algorithm correctly disambiguated 47% of the words to the sense level, and 72% to the homograph level.

Direct comparisons of these success rates with those of other methods is difficult. None of the other methods was used to disambiguate the same text, and while we have attempted to tag every ambiguous word in a sentence, other methods were applied to one, or at most a few, highly ambiguous words. It appears that in some cases the fact that our success rates include not only highly ambiguous words, but some words with only a few senses is offset by the fact that other researchers have used a broader definition of word sense. For example, the four senses of "interest" used by Zernick and Jacobs [1990] may correspond more closely to our two homographs and not our ten senses of "interest." Their success rate in tagging the three words "interest", "stock", and "bond" was 70%. Thus it appears that the method we propose is comparable in effectiveness to the other computational methods of word-sense disambiguation, and has the advantages of being automatically applicable to all the 28,000 words in LDOCE and of being computationally practical.

Below we give two examples of the results of the technique. The words following the arrow are the stemmed words selected from the definitions and used to calculate the redundancy. The headword and sense numbers are those used in the machine readable version of LDOCE.

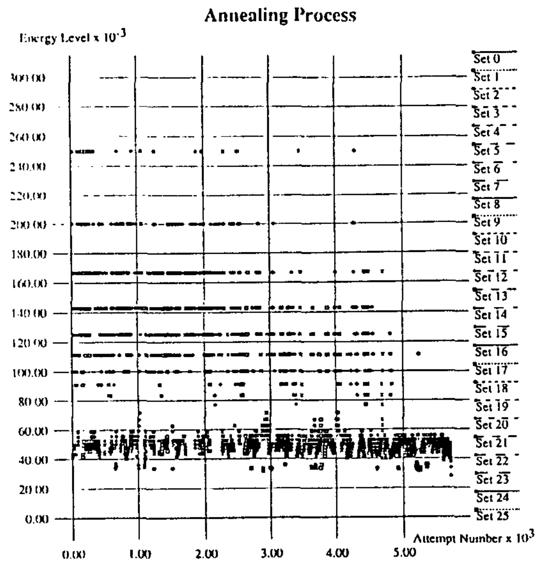
Finally, we show two graphs (figure ??) which illustrate the convergence of the simulated annealing technique to the minimum energy (E) level. The second graph is a close-up of the final cycles of the complete process shown in the first graph.

5. Conclusion

This paper describes a method for word-sense disambiguation based on the simple technique of choosing senses of the words in a sentence so that their definitions in a machine readable dictionary have the most words in common. The amount of computation necessary to find this optimal choice exactly quickly becomes prohibitive as the number of ambiguous words and the number of senses increase. The computational technique of simulated annealing allows a good approximation to be computed quickly. Advantages of this technique over previous work are that all the words in a sentence are disambiguated simultaneously, in a reasonable time, and automatically (with no hand disambiguation of training text). Results using this technique are comparable to other computational techniques and enhancements in-

```
SENTENCE
The fish floundered on the river bank,
struggling to breathe

DISAMBIGUATION

fish hw 1 sense 1 :
DEF -> fish creature whose blood change temperature
according around live water use its FIN  tail swim
river hw 0 sense 1 :
DEF -> river wide nature stream water flow between
bank lake another sea
bank hw 1 sense 1 :
DEF -> bank land along side river lake
struggle hw 1 sense 0 :
DEF -> struggle violent move fight against thing
breathe hw 0 sense 2 :
DEF -> breathe light live


SENTENCE
The interest on my bank account accrued over
the years

DISAMBIGUATION

interest hw 1 sense 6 :
DEF -> interest money paid use
bank hw 4 sense 1 :
DEF -> bank place money keep paid demand
where related activity go
account hw 1 sense 5 :
DEF -> account record state money receive paid
bank busy particular period date
accrue hw 0 sense 0 :
DEF -> accrue become big more addition
year hw 0 sense 3 :
DEF -> year period 365 day measure any point
```

Table 1: Sample Disambiguations



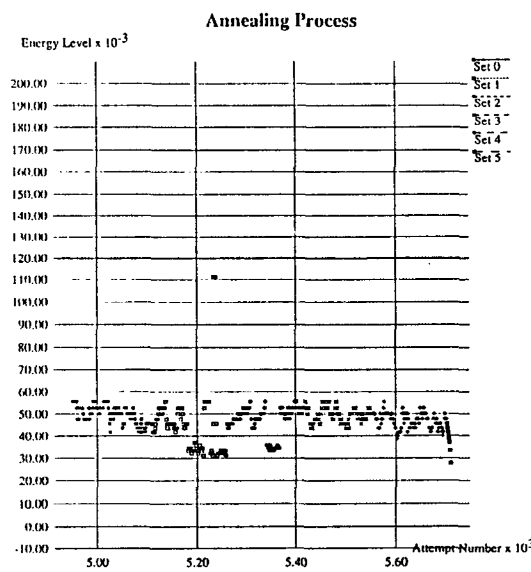


Figure 2:

corporating co-occurrence, part-of-speech, and subject code information, which have been exploited in one-word-at-a-time techniques, may be expected to improve the performance.

References

1. Lesk, Michael E. (1986). "Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone". *Proceedings of the ACM SIGDOC Conference*, Toronto, Ontario.
2. McDonald, J. E., Plate, T, and Schvaneveldt, R. W. (1990). "Using Pathfinder to Extract Semantic Information from Text". In Schvaneveldt, R. W. (ed.) *Pathfinder Associative Networks: Studies in Knowledge Organisation*, Norwood, NJ: Ablex.
3. Wilks, Yorick A., Dan C. Fass, Cheng-Ming Guo, James E. McDonald, Tony Plate, and Brian M. Slator (1990). "Providing Machine Tractable Dictionary Tools". *Computers and Translation 2*. Also to appear in Theoretical and Computational Issues in Lexical Semantics (TCILS). Edited by James Pustejovsky. Cambridge, MA: MIT Press.

4. Zernik, Uri, and Paul Jacobs (1990). "Tagging for learning: Collecting thematic relations from corpus". *Proceedings of the 13th International Conference on Computational Linguistics (COLING-90)*, Helsinki, Finland, 1, pp. 34-37.

5. Guthrie, J., Guthrie, L., Wilks, Y., and Aidinejad, H. (1991). "Subject-Dependent Co-Occurrence and Word Sense Disambiguation", *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, Berkeley, CA. June 1991. pp. 146-152. Also Memoranda in Computer and Cognitive Science MCCS-91-206, Computing Research Laboratory, New Mexico State University.

6. Hearst, M. (1991). "Toward Noun Homonym Disambiguation - Using Local Context in Large Text Corpora", *Proceedings of the Seventh Annual Conference of the UW Centre for the New OED and Text Research, Using Corpora* pp. 1-22.

7. Procter, P., R. Ilson, J. Ayto, et al. (1978) *Longman Dictionary of Contemporary English.* Harlow, UK: Longman Group Limited.

8. Wilks, Yorick A. (1975a). "An Intelligent Analyzer and Understander of English". *Communications of the ACM*, 18, 5, pp. 264-274. Reprinted in Readings in Natural Language Processing, Edited by Barbara J. Grosz, Karen Sparck-Jones and Bonnie Lynn Webber, Los Altos: Morgan Kaufmann, 1986, pp. 193-203.

9. Wilks, Yorick A. (1975b). "A Preferential Pattern-Seeking Semantics for Natural Language Inference". *Artificial Intelligence* , 6, pp. 53-74.

10. Wilks, Y. and Fass. D. (1991). "Preference Semantics: a family history", To appear in *Computing and Mathematics with Applications* (in press). A shorter version in the second edition of the Encyclopedia of Artificial Intelligence .

11. Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953) J. Chem. Phys. vol. 21, p.1087.

12. Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Science vol. 220, pp. 671-680.

13. Sampson, G. (1986). "A Stochastic Approach to Parsing". *11th International Conference on Computational Linguistics (COLING-86).* pp. 151-155.