A Stochastic Approach to Parsing

Geoffrey Sampson

Department of Linguistics & Phonetics
University of Leeds

1. Simulated annealing (e.g. Kirkpatrick et al. 1983, Bridle & Moore 1984, Ackley et al. 1985) is a stochastic computational technique for finding optimal solutions to combinatorial problems for which the combinatorial explosion phenomenon rules out the possibility of systematically examining each alternative. It is currently being applied to the practical problem of optimizing the physical design of computer circuitry, and to the theoretical problems of resolving patterns of auditory and visual stimulation into meaningful arrangements of phonemes and three-dimensional objects. Grammatical parsing -- resolving unanalysed linear sequences of words into meaningful grammatical structures -- can be regarded as a perception problem logically analogous to those just cited, and simulated annealing holds great promise as a parsing technique.

Simulated annealing can most directly be explained via a physical analogy. Consider the logical space of alternatives in some large-scale combinatorial problem as a chunk of mountainous terrain, in which the altitude of any point corresponds to the relative "goodness" of that particular solution (the lower, the better). We want to find the lowest point, but there are far too many points for each to be considered separately. We might try to locate the low point by dropping a ball onto the territory at random and hoping it will roll down to the low point. This corresponds to randomly choosing a particular overall solution to the combinatorial problem, and then considering a series of modifications to individual components of the solution, adopting the modifications whenever they improve the overall solution and rejecting them otherwise. But the ball is very unlikely to reach the low point. Much more probably, it will roll a short way downhill and come to rest in a "local minimum", a place where all immediate moves are uphill even though, some distance away, there are places much lower than the spot where the ball has halted.

In this situation, a good way of improving the search technique would be to pick up the landscape and shake it, so that the ball does not invariably roll downhill but sometimes bounces over obstructions. One would begin by shaking hard, so that even the highest peaks can be cleared, and then gradually reduce the amplitude of shaking so that the ball searches for lowness in terms of progressively finer detail. In computational terms, rather than deciding whether to adopt each of a series of modifications to the randomly-chosen initial position simply by reference to whether it yields a gain or a loss, one decides by reference to whether it yields a loss that is greater than a number whose magnitude tends to decrease as the process continues. This is simulated annealing.

Not all combinatorial problems are amenable to the annealing technique. If the desired low point in the mountainous terrain of the analogy happened to be at the bottom of a deep, narrow mineshaft sunk from a high place, annealing would not help to find it. But the logical geometry of many real-life combinatorial phenomena is more like the geometry of natural mountains, where there is a strong tendency for relatively low-lying points to be adjacent to many other relatively low-lying points. For such phenomena, simulated annealing can be an efficient way of arriving at optimal solutions.

2. The applicability of annealing as a parsing technique presupposes a statistical approach to NL analysis which will itself be unfamiliar to many readers. At this point I must therefore digress from the annealing concept in order briefly to describe the statistics-based NL research paradigm within which I am working, and to which simulated annealing appears to offer an important contribution.

Much current work in parsing, as represented in books such as King (1983), Jones & Wilks (1983), analyses input by systematically checking its properties against the various possibilities allowed by a grammar which specifies the language in question as a well-defined class of sentences. The grammar may be in the form of an ATN, a GPSG, or in some other form, and the checking process may operate in very diverse ways; but all these approaches have in common the notion of a clearcut boundary between a (probably very rich and complex) class of well-formed inputs, and other inputs which simply do not belong to the language under analysis. There are major difficulties in making such parsers work adequately with authentic, unedited input drawn from unrestricted domains. NLs are so endlessly diverse and unpredictable in the turns of phrase they display that many people find it extremely difficult to believe that any sets of rules, no matter how complex, can fully define them. It

remains as true as it was sixty years ago that "All grammars leak" (Sapir 1921: 38).

The Unit for Computer Research on the English Language (UCREL) of the University of Lancaster, led by Geoffrey Leech and Roger Garside, with whom I remain associated since my recent move to Leeds, has made considerable progress in recent years in developing automatic routines which succeed in analysing authentic text using techniques which do not assume the existence of a clearcut grammatical/ungrammatical distinction (cf. Garside et al.: forthcoming). The first major UCREL achievement was the CLAWS word-tagging system (see e.g. Atwell et al. 1984). Since 1982 CLAWS has been assigning part-of-speech tags, drawn from a finely-differentiated, 134-member tag-set, to words of authentic running English text (which are often grammatically many ways ambiguous in isolation), with a consistent accuracy level of 96-97% of words correctly tagged -- a figure which we believe can be further improved by tuning the system in various ways. This is an achievement to which we have been unable, despite extensive enquiries, to discover near rivals.

The significant point about CLAWS is that it embodies no knowledge whatever of the overall grammatical architecture of English sentences. Instead, it uses an empirically-derived matrix of relative transition-probabilities between pairs of adjacent tags, together with information enabling a set of candidate tags to be identified for any given word taken in isolation (using rules which refer to the last letters of the word's spelling, together with a list of c. 7200 exceptions). Simplifying greatly, CLAWS works by forming all possible paths through sequences of tags which are candidates for the words of a sentence, and choosing the path for which the product of the successive transition probabilities is highest. As a matter of policy, no entries in the matrix of tag-transition probabilities are zero; we know as well as other linguists that failure to observe a particular transition in our data does not imply that the transition is "ungrammatical", and therefore even unobserved transitions are assigned a small positive probability. Thus it is true to say that the system "knows" nothing about English in the sense of drawing sharp distinctions between grammatical and ungrammatical sequences; it deals exclusively in relative likelihoods. Yet this wholly "unintelligent" system works extremely well. It is easy to make CLAWS fail by inputting "trick" sentences of the kind often encountered in linguistics textbooks, but the lesson of CLAWS is that such sentences are notably rare in real life.

We are currently developing a CLAWS-like solution to the harder problem of grammatical parsing. We have built up a database of manually-parsed sentences, from which we extract statistics that allow a likelihood measure to be determined for any logically possible non-leaf constituent of a parse-tree. That is, given a pairing of a mother-label with a sequence of daughter-labels, say the pair ‹J, NN JJ P›, the likelihood function will return a figure for the relative frequency with which (in this case) an adjective phrase consists of singular common noun + adjective + prepositional phrase. (In the case quoted the likelihood will probably be low, but it ought not to be zero: I selected the example after encountering, in a book opened at random, the adjective phrase underlined in "the value obtained must be assignment compatible with the type of the variable ...".) We assume, I believe with justification, that with only minor special provisos the likelihood of a full parse-tree can be identified with a simple function of the likelihood of each of its non-leaf nodes.

3. The most direct way to imitate the CLAWS technique in the parsing domain would be to generate all possible tree-structures for a given sentence taken as a sequence of word-tags, and all possible labellings of each of those structures, and choose the tree whose overall plausibility figure is highest. Unlike in the case of word-tagging, however, for parsing this approach is wholly impractical. The average sentence in our database is about 22 words long, and the set of nonterminal symbols recognized by our parsing scheme has almost thirty members; the number of alternative logically-possible labelled tree structures having 22 terminal nodes is astronomical. I have therefore begun to experiment with simulated annealing as a solution to the problem. The grammatical statistics in the experiment described here are far cruder than would be needed for a full-scale annealing parser, but initial results are nevertheless promising.

4. To apply the annealing technique to the parsing problem, it is necessary: (i) to state a tree-evaluation function; (ii) to define a class of local changes to trees, such that any logically-possible tree can be converted to any other by applying a series of changes drawn from the class (we cannot allow the initial randomly-chosen tree to eliminate the possibility of ever reaching some other tree which might be the correct one); and (iii) to define an annealing schedule.

Tree-evaluation in my experiment is based on statistics of constituent-daughter transition frequencies: a constituent labelled A and having daughters labelled B C D is given a value derived from the observed frequencies of the transitions A/[B, A/BC, A/CD, A/D]. (The functions which derive node values from daughter-transition frequencies, and tree

values from node values, are more complex than simple averaging, which is unsatisfactory because it too easily allows an individual "bad" value in a candidate parse-tree to be offset by several "good" values elsewhere in the tree. I do not give details of the functions currently used.)

In the experiment, the statistics referred to a very small set of broadly-defined node-labels, comprising 14 nonterminal labels and 30 word-class labels. Our database uses a parsing scheme which recognizes distinctions much finer than this -- we have seen that 134 word-classes are distinguished, and nonterminal labels can include subcategory symbols which in theory permit on the order of 60,000 distinct labels. However, most of this information was discarded for the sake of simplifying the pilot experiment.

For point (ii) above, I define a possible move as follows. Given a parse-tree, select a node other than the root at random. Disconnect it from its mother. It will then be located within an "arch" of nodes whose left and right bases are respectively the last word before and the first word after the disconnected constituent, and whose "keystone" is the lowest node dominating both of those words. Choose at random either any node located on the arch other than the two bases, or any link between two nodes in the arch. In the former case, attach the disconnected node to the chosen node as an extra daughter, and relabel the new mother of the disconnected node with a randomly-chosen label. In the latter case, create a new node on the chosen link between existing nodes, label it with a randomly-chosen label, and attach the disconnected node to it as a sister of the node at the lower end of the chosen link. In either case, if the ex-mother of the disconnected node is left with only one daughter, then delete the ex-mother by merging its upward and downward links (if the ex-mother is the root node, its remaining daughter becomes the new root and is accordingly relabelled S). It is easy to show that any tree can be derived from any other tree via a series of moves of this kind.

There is no "magic formula" to determine the ideal annealing schedule for a given class of combinatorial optimization problems: this depends on the geometry of the logical space of possibilities, and has to be discovered by experiment. The requirements are that annealing must begin at a high enough "temperature" for the system to be thoroughly "melted" (that is, the factor by which the negativity of locally-negative moves is discounted must be sufficiently large for moves to occur at random with no significant bias towards locally-positive moves), and "cooling" must take place slowly enough for adequate searching of the possibility-space to occur. (If the initial temperature is unnecessarily high, or cooling unnecessarily protracted, a penalty will be paid in extra processing for little or no gain in ultimate accuracy of search.) "Temperature" might be treated as a constant figure which is added to the result of subtracting previous likelihood-value from subsequent likelihood-value in determining whether a move under consideration yields a net gain and is therefore adopted. What is usual, however, is to strengthen the analogy with thermodynamics by adding, not a constant figure, but a figure drawn randomly from a Gaussian distribution, with "temperature" standing for the standard deviation of the distribution. Thus, even at a high temperature it will sometimes happen that a slightly locally-negative move is rejected, and even at a low temperature it will occasionally happen that a strongly locally-negative move is accepted. (Locally-positive and neutral moves are always accepted at any temperature.)

5. Let me illustrate by quoting one of the first annealing runs carried out by the system, on a short sentence input as

d j j n o v i d j n .

Brief glosses for these symbols are: d, determiner; j, adjective; n, singular noun; o, modal verb or do; v, main verb; i, preposition; ., sentence-final punctuation mark. Thus the sequence stands for a sentence such as The quick brown fox will jump over the lazy dog. This is of course an artificially simple example, and if authentic language were commonly as "well-behaved" as this then the case for using stochastic parsing techniques would be weak. However, notice that the technique embodies no concept of a contrast between well-formed and deviant strings, so that in principle it should be as easy to set an annealing parser to seek the "least implausible" analysis of a highly deviant input as to seek the correct analysis of a thoroughly well-formed input. The reason for beginning with a simple example is that I anticipate that the performance of the system will become more sensitive to details of the evaluation function and annealing schedule as inputs become more complex and less well-formed, and at present I have only begun to explore alternative evaluation functions and annealing schedules.

The schedule used for the run to be illustrated was as follows. The structure initially assigned to the string was the "flat" structure in which each word is an immediate constituent of the root: [S djjnovidjn.] This tree is assigned the value -2.26 by the tree-evaluation function. The initial temperature (standard deviation of the Gaussian) was 1. The temperature was reduced by 3% after every fiftieth successive attempt to change the tree. The system was deemed to have "frozen" at the first temperature-drop at which each of the 100 preceding attempts to change the tree either had been rejected or left the value of the tree unchanged.

To give the reader a feeling for the way an annealing run proceeds, I display the situation reached after every hundredth attempted tree-change. On each line I show the temperature reached immediately before the drop which occurs at that point, the proportion of the last hundred attempted changes which were accepted, the value of the current tree, and the tree itself. Nonterminal symbols are represented by capitals written immediately after the opening bracket of the constituent they label; closing brackets are unlabelled.

Attempts	Temp.	Changes	Value	Current Tree
100	0.970	93	-1.31	[Sdjj[Rn[Lo[G[Fvi][J[N[Gdj]n].]]]]]
200	0.913	93	-1.52	[S[Wd[D[Jjj]no]][Fv[Vidj]][Pn.]]
300	0.859	92	-1.02	[S[Pd[Dj[Jjn]][Nov]i[R[Adj]n]].]
400	0.808	88	-2.30	[Sd[Dj[V[Fjn][P[V[Gov][Ji[Ldj]]n].]]]]
500	0.760	82	0.00976	[S[Vdj][Njn[W[Tovidj]n]].]
600	0.715	90	-0.536	[S[N[Jd[T[Pjj][S[N[Tn[Jov]][Tid]]j]]]n].]
700	0.673	66	1.64	[S[Sd[Njj[Nn[No[Vvi]]]d[Njn]]].]
800	0.633	69	-1.51	[S[N[Nd[Sjj]]no][J[A[Tvi][Tdjn]].]]
900	0.596	73	-0.0562	[S[N[D[F[Ad[S[Fjj][Nno]]v][Tid]j]n].]
1000	0.561	75	-0.984	[Sd[D[L[Gj[Njn]][Wov]][F[F[Lid]jn].]]]
1100	0.527	58	1.50	[S[P[Mdj][Njn][Jovi][Ndj]n].]
1200	0.496	66	-0.184	[S[Ndjjn][Novid]j[Mn.]]
1300	0.467	63	0.668	[S[N[Dd[Njj]n]ov[Lidj]n].]
1400	0.439	54	0.325	[S[Nd[Njj]n][P[Jo[Rvi[Sdj]]n].]]
1500	0.413	57	0.760	[S[Nd[V[Njj]n]o][S[T[Tvi][Dd[Njn]]].]]
1600	0.389	18	4.93	[S[Ndjjn[F[Vov][Pid]]][Njn].]
1700	0.366	8	5.21	[S[Ndjjn][F[Vov][Pi[Nd[Njn]]]].]
1800	0.344	11	5.46	[S[N[Ndjjn][F[Vov][Pi[Ndjn]]]].]
1900	0.324	7	5.70	[S[Ndjjn][F[Vov][Pi[Ndjn]]].]
2000	0.305	5	6.63	[S[Ndjjn][Vov][Pi[Ndjn]].]
2100	0.287	5	6.63	[S[Ndjjn][Vov][Pi[Ndjn]].]

On this run the system froze at temperature 0.287, after 2100 attempted changes of which 1173 were accepted. The structure attained at freezing is the correct structure for the input sequence, according to our parsing scheme. (The symbols N, P, V stand for noun phrase, prepositional phrase, and verb phrase -- the latter in our terms referring to a sequence of auxiliary and main verbs, not including object, complement, etc. We recognize no internal structure in a noun phrase such as the quick brown fox.)

Not all runs of this pilot system have been as completely successful as this, though none have frozen on totally crazy trees. Yet the range of possibilities out of which the system has winnowed the correct analysis includes massive numbers of utterly crazy structures: note for instance how in the early stages of the run illustrated the system has considered a tree including a genitive phrase (G) consisting of a finite clause (F) followed by an adjective phrase (J) -- a constituent which linguistically makes no sense at all. Considering how many alternative logically-possible solutions are available to the system, a few thousand steps seems a small number by which to reach the correct solution or even its vicinity. Although at present some mistakes are made, there is plenty of scope for improving performance by refining the grammatical statistics and evaluation function, and modifying the annealing schedule. At this admittedly very early stage I regard the prospects for parsing by annealing as highly promising.

6. Simulated annealing appeals strongly to some writers (e.g. Bridle & Moore 1984: 315) as a model of psychological perception mechanisms. In the case of grammatical parsing, though, there is one respect in which the model presented so far is quite implausible psychologically: it ignores the left-to-right sequential manner in which humans process written as well as spoken language. There is a natural way to incorporate time into an annealing parser which not only is psychologically plausible but promises greatly to increase its efficiency as a practical automatic system.

Rather than a whole sentence being submitted to the annealing system at once, in a "dynamic" annealing system parsing would proceed in a series of rounds. The input to the nth round would be an annealed parsing of the first n-1 words of the sentence, followed by the nth word; annealing would begin anew at melting temperature on this input. The opportunity for efficiency would arise from the fact that NL grammar only rarely forces the reader to backtrack -- the insight on which Mitchell Marcus's Parsifal system was founded (Marcus 1980). Marcus's strategy involved a total exclusion of backtracking from his central parsing system, with "garden path" sentences being handed over to a quite separate "higher level problem solver" for processing. However, Marcus's predictions about a sharp categorization of NL sentences into garden-paths and non-garden-paths have provoked considerable criticism. In a dynamic annealing parser, all parts of the current tree would at all

stages be available to revision, but the relative rarity of the need for backtracking could be exploited by adding a bias to the function which randomly selects nodes for reconsideration, so that nodes are reconsidered less frequently as they become "older". Since the bulk of computing time in an annealing parser would undoubtedly be consumed in calculating gains and losses for candidate tree-changes, this system of concentrating the search for profitable tree-changes on the areas of the tree where such changes are most likely to be found could be a good means of saving processing time by reducing the total number of moves considered.

7. A problem that will not have escaped the reader's attention is that I have discussed parsing purely in terms of finding surface parse-trees (which happens to be the task which the UCREL group are engaged on). It is not obvious how to extend the annealing approach so as to yield deep parses. However, there is nothing about simulated annealing that makes it intrinsically inapplicable to the task of deep parsing. What needs to be done is to define a class of logically-possible deep parse-trees and a class of moves between them, and to find an evaluation function which takes any pairing of a deep structure with a surface word-sequence into a likelihood-value. This task is very different in kind from the work currently done by theoretical linguists and AI researchers interested in underlying or logical grammar, who tend to have little time for statistical thinking, but that is not to say that the task is necessarily senseless or impossible. Deep parsing, if possible at all, will presumably need to exploit semantic/"inferencing" considerations as well as information about grammar in the narrow sense, but nothing says that these matters might not be built into the evaluation function.

8. Finally, it may be that annealing is useless as a parsing technique because the geometry of NL parsing space is wrong. Perhaps the space of English parse-trees (whether surface or deep) resembles the Witwatersrand rather than the Cotswolds, being an upland plateau riddled with deep goldmines rather than a rolling landscape whose treasures lie exposed in valley bottoms. I conjecture that NLs are Cotswold-like rather than Rand-like, and that, if they were not, humans could not understand them. Only empirical research using authentic data can settle the question.

REFERENCES

Ackley, D.H., G.E. Hinton, & T.J. Sejnowski 1985 "A learning algorithm for Boltzmann machines". Cognitive Science 9.147-69.

Atwell, E.S., G.N. Leech, & R.G. Garside 1984 "Analysis of the LOB Corpus: progress and prospects". In J. Aarts & W. Meijs, eds., Corpus Linguistics. Rodopi.

Bridle, J.S. & R.K. Moore 1984 "Boltzmann machines for speech pattern processing". Proceedings of the Institute of Acoustics vol. 6 pt. 4 pp. 315-22.

Garside, R.G., G.N. Leech, & G.R. Sampson, eds. Forthcoming. The Computational Analysis of English. Longman.

Jones, K.S. & Y.A. Wilks, eds. 1983 Automatic Natural Language Parsing. Ellis Horwood.

King, M., ed. 1983 Parsing Natural Language. Academic Press.

Kirkpatrick, S., C.D. Gelatt, & M.P. Vecchi 1983 "Optimization by simulated annealing". Science 220.671-80.

Marcus, M.P. 1980 A Theory of Syntactic Recognition for Natural Language. MIT Press.

Sapir, E. 1921. Language. Harcourt, Brace & World.