A Rule-Based Connectionist Parsing System

Bart Selman and Graeme Hirst
Department of Computer Science
University of Toronto
Toronto, Canada M5S 1A4

Abstract

We describe a connectionist parsing scheme based on context-free grammar rules. In this scheme we use an updating rule similar to the one used in the Boltzmann machine (Fahlman, Hinton and Sejnowski 1983) and apply simulated annealing. We show that at low temperatures the time average of the visited states at thermal equilibrium represents the correct parse of the input sentence.

In contrast with previously proposed connectionist schemes for natural language processing, this scheme handles the traditionally sequential rule-based parsing in a general manner in the network. Another difference is the use of the computational scheme of the Boltzmann machine. This allows us to formulate general rules for the setting of weights and thresholds in our system.

The parsing scheme is built from a small set of *connectionist primitives* that represent the grammar rules. These primitives are linked together using pairs of computing units that behave like discrete switches. These units are used as binders between concepts represented in the network. They can be linked in such a way that individual rules can be selected from a collection of rules, and are very useful in the construction of connectionist schemes for any form of rule-based processing.

1. Introduction

Recently, several connectionist models for natural language understanding (NLU) have been proposed; for example, Waltz and Pollack (1984; Pollack and Waltz 1982) and Cottrell and Small (1983; Small, Cottrell, and Shastri 1982) give models for word-sense and syntactic disambiguation, and Reilly (1984) gives a scheme for anaphora resolution. The models are based on the deterministic connectionist scheme (McClelland and Rumelhart 1981; Feldman and Ballard 1982). A central aspect of these schemes is that they process the different sources of knowledge used in NLU, such as lexical and world knowledge in a highly integrated way; for example, the syntactic and semantic processing are combined.

A major limitation of these schemes is their very limited capability to handle tasks such as parsing and case filling which seem to require processing to be based on a set of rules. For example, Pollack and Waltz use the output of a conventional chart parser to generate a network for the syntactic parse of the sentence. This network only represents the parse tree (or trees, in case of syntactic ambiguity) of the particular input sentence. We propose a more general approach, namely a network that directly represents the grammar rules and is to be used for parsing of a large number of sentences. A similar approach could be used for other types of rule based-processing, like case filling.

This work was supported by a Government of Canada Award to the first author, and grants from the University of Toronto and the Natural Sciences and Engineering Research Council of Canada to the second author.

Our motivation behind this research is twofold. On one hand we believe that, at least part of the natural language understanding process can be handled by a connectionist architecture and that this form of integrated, parallel processing facilitates the parsing process. On the other hand, we believe that these schemes can only be of practical interest for NLU if they handle rule-based processing, like syntactic parsing, in a general and efficient way and are understood well enough to set the weights and thresholds correctly in large networks.

2. The model

2.1 Topology

We base our scheme on a context-free grammar, but this is not essential in our approach. The syntactic categories of the grammar are represented in a localist manner, that is, each syntactic category is represented by a unit in the network. As we will see this localist approach allows us to represent the grammar rules in a very straightforward manner, and consequently determines the set of non-zero weights (giving the topology of the network). The grammar rules will determine how these units are interconnected.

In the scheme we distinguish two layers. The *input layer* consists of a number of computing units representing the terminal symbols of the grammar. An input sentence will activate some subset of these units. Connected to this input layer is a network that represents the parse trees of all non-terminal strings in the language whose length is not greater than the number of units in the input layer. This network, called the *parsing layer*, is constructed from *connectionist primitives*, which represent the context-free grammar rules. Figure 1 gives two examples of such primitives. The activation of all units of a primitive corresponds to the use of the associated grammar rule in the parse. The number of units in the parsing layer depends on the particular context-free grammar rules and the number of input units. The different parse trees are not represented by completely disjoint sets of units, but share common substructures. This will keep the size of the network manageable.

We use intermediate computing units to link the primitives together. These units play the role of binders in the network and, therefore called *binder units*[1]. The computing units representing the terminals and variables of the grammar are called *main units*.

Figure 2 gives an example of the use of binder units. The four binder units are used to represent the fact that the main unit #0 is part of three grammar rules:

$$VP \rightarrow VP \;\; PP \qquad \text{(1a)}$$

$$VP \rightarrow verb \qquad \text{(1b)}$$

$$VP \rightarrow verb \;\; NP \;\; . \qquad \text{(1c)}$$

The binders are linked in such a way, using inhibitory and excitatory connections, that when the network reaches a stable state the active binders (those whose output equals +1) tell us which one of the three possible grammar rules is used in the parse of the input sentence to decompose the verb phrase represented by unit #0. So, if binder #1 stays active rule 1a is used in the parse, if binder #2 and #3 stay active rule 1b is used and if binder #2 and #4 stay active rule 1c is used.

2.2 Computational scheme

In this section we will consider the way in which the network finds the parse of a sentence. In the input layer of the network, the units are placed in *input groups*. Each group contains a unit for each terminal symbol of the grammar. The input groups are numbered; the n^{th} group is associated with the n^{th} word in the input sentence. Initially the computing units of both the input and the parsing layer of the network are inactive (their output is –1).

[1] Binder units with a similar function are used by Cottrell (1985) in his parsing system based on a deterministic connectionist model.

As a sentence comes in, each word of the sentence activates the computing unit(s) representing its associated syntactic category or categories. So the first word of the sentence activates one or more units (depending on the number of syntactic categories associated with the word) in input group #1, the second word one or more units in input unit #2, and so on. After receiving input data, the network starts the relaxation process. During this process, the outputs of the activated computation units in the input groups are fixed at +1, while the outputs of other units in the input layer are fixed at -1, so that the network can find the optimal match between the input data and the internal constraints representing the grammar rules; this match will represent the correct parse of the input.

In our model we use a variation on the computational scheme of the Boltzmann machine (Fahlman, Hinton and Sejnowski 1983; Hinton and Sejnowski 1983a, 1983b), and apply the simulated annealing scheme of Kirkpatrick et al. (1983) to find the optimal match between input data and internal constraints. Our scheme differs from the original in that we use -1 and +1 as output values of our computing units instead of 0 and +1. This facilitates the representation of symmetrical interdependency relations between hypotheses in the scheme; there exists a one-to-one mapping between this scheme and the original (Selman 1985).

The fact that this scheme searches for a global energy minimum and that at equilibrium the relative probability of a particular state of the system is given by its energy enables us to formulate general rules for the setting of the weights on the connections and the thresholds of the computing units.

We compute the average value of the output of each unit at the different temperatures used in the annealing scheme. In an example given below, we will see how these average values will change during cooling of the system; finally, at a temperature just above the freezing point of the system, the units with outputs close to +1 will represent the parse of the sentence. To find the temperature just above the freezing point of the network, we consider statistical data on the behavior of the network during simulated annealing.

2.3 The setting of weights and thresholds

The setting of weights and thresholds is probably the most difficult problem in the design of a connectionist scheme. The set of weights and thresholds represents the internal constraints and therefore the knowledge in the system. So far we have described how units are interconnected in our parsing scheme; that is the set of links with non-zero weights. Now we will discuss what values should be chosen for the weights on these links.

In the Boltzmann formalism, the behavior of the system during relaxation can be described as a search for a global minimum in the energy of the network

$$E = \sum_k E_{loc,k} \qquad (2)$$

In which

$$E_{loc,k} = (-1/2 \sum_j w_{kj} s_j + \theta_k) s_k \qquad (3)$$

is the contribution of the k^{th} unit with output value s_k and threshold θ_k to the energy, w_{kj} is the weight on the connection between the k^{th} and the j^{th} unit (we assume symmetrical connections), and the summation in equations (2) and (3) is over all units in the network.

Given the fact that the network searches for a global energy minimum, we can, to a first approximation, analyze the behavior of the network by assuming that each unit and its direct neighbors will chose output values such that $E_{loc,k}$ becomes minimal. However this method gives only a rough approximation of the actual behavior, because minimizing E_{loc} for one particular unit often conflicts with minimizing E_{loc} of other units. To get a better insight in the behavior of the system we therefore consider the contribution to the global energy of a

small groups of units.[1] Because of the homogeneous structure of our network we only have to consider a limited number of cases. As an example we will consider the setting of the weights on the excitatory links.

Figure 3 shows some excitatory links in a typical configuration. The network represents two grammar rules:

$$VP \rightarrow verb \qquad (4a)$$
$$VP \rightarrow verb\ \ NP \qquad (4b)$$

Rule 4a is represented by the units 0, 1, and 3; rule 4.4b by the units 0, 2, 4, and 5. During the relaxation process our network has to decide between rule 4a and rule 4b or neither of them. There is no a priori preference for one rule over the other. Therefore, because unit #2 is connected to two other units representing the left-hand side of grammar rule 4b and unit #1 is connected to only one unit representing the left-hand side of grammar rule 4a we have to make the weight on the link between units #1 and #3 twice as strong as the links between units #2 and #4 and between units #2 and #5. (This can be easily generalized for grammar rules with more symbols; one choses the weights such that the sum of the inputs at the binder units is equal for all grammar rules.) So we choose $w_{2,4}$ and $w_{2,5}$ equal to some positive constant and we set $w_{1,3}$ to twice this constant. We set this constant to 1.0. One should note that the absolute value of the constant is irrelevant. This value is only going to to determine at what temperature in our simulated annealing scheme the system is going to freeze, but the temperature is only a formal parameter introduced to do simulated annealing and has no meaning in our final result.

For the use of a grammar rule in the parse, the presence of each symbol in the rule is equally important, and therefore we connect the units in a connectionist primitive representing a grammar rule with links of equal strength, so $w_{4,5}= 1.0$. And finally, because bottom-up and top-down parsing is completely integrated and of equal importance in our networks we choose $w_{0,1}=w_{0,2}= 2.0$.

Selman (1985) gives similar analyses that lead to rules for for setting of weights on inhibitory links and the thresholds of the units. Here is a summary of these rules:[2]

$weight_{excitatory\ link}$	+1.0	*in primitive with three units*	(5a)
	+2.0	*in primitive with two units*	(5b)
$weight_{inhibitory\ link}$	–3.0		(5c)
threshold	0.0	*main unit*	(5d)
	–2.0	*main unit in symmetrical environment*	(5e)
	+2.0	*binder unit.*	(5f)

A *main unit in a symmetrical environment* is a main unit only linked to pairs of binder units (that is connected to both binders) and at most one other binder unit.

Although the local analyses and symmetry considerations on which these rules are based won't guarantee the right global behavior, good simulation results of a network with weights set according to these rules show that apparently such a local analysis gives a reasonable estimate of the global behavior of the parsing network. This is most presumably a consequence of the highly homogeneous structure of our parsing scheme (the networks are built from a small number of primitives).

[1] Of course, for an exact analysis one would have to consider all possible states of the (total) network; this becomes clearly infeasible for networks with more than about 25 nodes.

[2] Often only the ratio of between the different weights and thresholds are relevant; Selman (1985) gives these rules in a more general form.

3. The design and testing of a network

To illustrate our model we will now consider an example. This network is based on the following context free grammar rules (taken from an example in Winograd 1983):

$$
\begin{array}{lll}
S \rightarrow NP\ VP & NP \rightarrow determiner\ NP2 & \\
S \rightarrow VP & NP \rightarrow NP2 & \\
VP \rightarrow verb & NP \rightarrow NP\ PP & (6) \\
VP \rightarrow verb\ NP & NP2 \rightarrow noun & \\
VP \rightarrow VP\ PP & NP2 \rightarrow adjective\ NP2 & \\
PP \rightarrow preposition\ NP & &
\end{array}
$$

We will represent five input groups; in a complete network each input group has a unit for all terminals of the grammar, however to make our example network less complex, we will not represent each terminal in each input group.

For the grammar rules in (6) we can construct connectionist primitives similar to those given in figure 1. To build the parsing layer upon the input layer, using these primitives, we consider the different possible ways in which the syntactic categories can be grouped according to the grammar rules, and design a network that represents those possibilities. One way we could have proceeded is by designing a set of networks, each representing the parse of one unique input sentence, linking all these networks to the input layer, and placing inhibitory connections between them. These inhibitory connections should guarantee that after the parsing network is given an input sentence, only the sub-network representing the parse of the input would remain active.

Apart from the question of whether the design of such a network is even feasible, there are two fundamental reasons why we did not take this approach. Firstly, many parse trees have common sub-structures. So one can save computing units by representing a common sub-structure by one set of units and linking that structure, using binder units, to the different parse trees represented in the network. Secondly, main units represent general concepts, such as 'noun phrase'. It is unlikely that in the human brain (connectionist models are to a certain extent modeled after the brain) these concepts are represented at many different places. Therefore, instead of representing the same general concept at many places in the network, we try to limit the number of main units representing a concept.

So, instead of constructing a network from a set of separate networks, each representing the parse of a sentence, we take an approach in which we try to share common syntactic structures between parse trees and minimize the number of main units. Following these guidelines we can construct from the input layer, using the connectionist primitives, a network like the one given in figure 4.[1] The weights and thresholds in this scheme are set according to (5).

To demonstrate the parsing capability of our network we consider the input sentence[2]

$$noun\ \ verb\ \ preposition\ \ determiner\ \ noun\,, \qquad (7)$$

exemplified by "*John ran down the hill*". For this sentence we ran a simulation of the parallel network on a serial machine using a simulated annealing scheme. To apply this scheme, one has to choose a descending sequence of temperatures such that the system has a reasonable chance of finding the state with the global energy minimum. Therefore one starts at a high temperature and first cools rapidly; once the system approaches the freezing point (the point

[1] Some simple input sentences show that the multiple units for NP's, NP2's, and PP's are necessary; however, one could further minimize the number of VP's. However, this results in a network where the connectionist primitives are less visible, and which is therefore harder to understand.

[2] The network has been successfully tested for a number of input sentences, including some cases of syntactic ambiguity (in such cases more than one unit is activated within an input group), in which no semantic knowledge is necessary to resolve the ambiguity (Selman 1985).

at which it settles down in a state with a local or a global energy minimum; in this state the temperature is too low to escape from this minimum) one should cool very slowly. As we will see, we don't have to freeze the system completely; the right parse of the input sentence is found at a temperature just above freezing.

At each temperature above the freezing point one has to take sufficient computation steps to allow the system to reach equilibrium at that temperature.

To be able to choose the sequence of temperatures and the number of computation steps we did some test runs with the network. An appropriate sequence of temperatures was determined by considering the number of changes in the output value of each computing unit and the energy distribution of the system at each temperature. Based on these indicators we choose a sequence of temperatures starting at $T = 10000$ (to randomize the system), followed by $T = 4.0$, $T = 2.0$, and then in steps of 0.2 down to $T = 0.6$.

To estimate the required number of computation steps at each temperature, we considered the results of a sequence of simulations in which this number was slowly increased. When the average output values of the units become independent of the number of computation steps we assume that enough computation steps have been taken to scan the energy distribution of the system at equilibrium. Two thousand computation steps (i.e. 2000 updates of each unit) per temperature appeared to be sufficient. It should be noted that we did not try to minimize the number of temperatures and the number of steps per temperature.

Figure 5 shows the annealing process for sentence (7); we give six temperatures. Each panel shows the average output value of each computing unit at the temperature given below the panel. The numbers 0 to 44 are the numbers of the computing units, the vertical position indicates their average output value on the interval [-1,+1]. At $T \approx 1.0$ the system freezes; below this temperature the system stays in one state. Comparison of these results with the parse tree of this sentence given in figure 6 shows that the time average of the outputs of the units, at a low non-zero temperature, corresponds to the correct parse of the input sentence if one chooses the units with outputs close to +1 as being part of the parse tree. At low temperatures there is a clear distinction between units with output close to +1 and the other units, as can be seen in figure 5.

Although the set of average output values of the units in this section does not reveal any significant differences between the system in a frozen state or just above the freezing point; more information about the parse can be obtained at a temperature just above the freezing point, as we will see in the next section.

4. Changing weights and thresholds

The weights in the example network given above were set in accordance with the rules given in (5). Because the setting of the weights and thresholds is an important issue in connectionist models, we will now consider what happens if we change some of them. We use again the example network given in figure 4 and, input sentence (7).

First we set the threshold of main unit #32 equal to zero; originally this threshold was set to -2.0, following rule (5e). The simulation results show that in the frozen state the system gives the correct parse, except for the main nodes #18 and #32 and the binder #21; that is the average output values of those units are -1.0. However the average of the output values of these units at a temperature just above freezing are almost equal to zero. This means that at that temperature these units are part of the parse of the sentence for approximately 50% of the time (all other average output values are close to -1.0 and +1.0, consistent with the correct parse). This result can be explained as follows. Just above the freezing point the system jumps between two states, namely:

- state **a**, all units of the correct parse are active; and
- state **b**, same as state **a**, except for the units 18, 21, and 32.

States **a** and **b** have approximately the same low energy; however to jump between these states the system has to visit a state with a higher energy. At a temperature above the freezing temperature there is enough thermal energy to visit the intermediate state with a higher energy; in other words the system has a reasonable chance to visit in the intermediate state compared to the chance to be in the lower energy states **a**, and **b**. Therefore, the system will jump between state **a** and **b** and the average output value of the units 18, 21, and 32 will be around zero. However, when the temperature is lowered the system freezes, that is it will settle in one of the states **a** or **b**, and there is not enough thermal energy to jump to the other low-energy state.

This example clearly demonstrates that the average output values of the units give more relevant information when determined just above the freezing point of the system than at or below that point. It also demonstrates how the Boltzmann mechanism not only finds a global minimum in the energy, but just above the freezing point the system jumps between a number of states with energies close or equal to the global energy minimum. This is an important advantage over a deterministic scheme. Even in case such a scheme manages to find one of the states **a** or **b**, it is very unlikely that a deterministic scheme, just before finding **a** or **b**, would pass through the other state with minimal energy. This example also shows that if we don't follow all the rules given in (5), the system does not behave as well as when we do; however the model still comes up with a result close to the correct parse.

We will now consider what happens if we increase both the strength of the inhibitory links between binder units and the thresholds of these units. We choose a weight of -20.0 on the inhibitory links and thresholds of $+19.0$. These values are in accordance with the general rule for setting the thresholds on binder units and the weights on inhibitory links between them. In this case we don't find any significant differences between the simulation results with this new choice of weights and thresholds and those using the original values.

This is an interesting result, because with this choice of weights it becomes extremely unlikely, at low temperatures, to find a pair of binder units with both outputs equal to $+1.0$. Such a pair would give a large positive contribution to the energy; see equation (3). Therefore, the pairs are functioning as three-state switches with at most one unit with output equal to $+1.0$. This is useful during the search for a correct parse (or global energy minimum), because a pair with both outputs equal to $+1$ corresponds to the obviously incorrect situation in which two grammar rules are applied at the same time to decompose a syntactic category.

In figure 2 we saw two pairs of binder units linked in such a way that they can choose the application of one specific grammar rule out of three. Using a similar approach one can design a network from pairs of binder units that can select one rule out of a large collection; such networks will be useful in general connectionist schemes for rule-based processing.

5. Conclusions

We have discussed how traditional typically sequential, rule-based processing like parsing can be done in a completely parallel manner. In the design of such connectionist schemes the use of pairs of intermediate units that function as binders between units that represent the different concepts appears to be very useful. One can choose the thresholds of the binder units and the weights on the inhibitory links in between them such that they function as three-state switches (both units on is a 'forbidden' state). These pairs linked together in a binary tree structure can be used to select one rule (for example, a grammar rule) out of a collection of alternatives. During the search for an optimal match between input data and the internal constraints in the network, the binder pairs select different rules to test whether they should be used. Interestingly, this bears a close resemblance to how a sequential processing scheme tries rule after rule; the advantage of the connectionist scheme is that many rules, each part of a different collection and represented in different parts of the network, can be applied in parallel, and also there is a complete integration of bottom-up and top-down processing.

We saw that the special properties of the computational scheme of the Boltzmann machine made it possible to set the weights and thresholds by analyzing the energy of small groups of units and some general symmetry considerations. Another useful aspect of the Boltzmann scheme is that the network at temperatures just above the freezing points visits a number of states with energies equal or close to the global energy minimum of the network. Such states will, in general, show only minor differences from the state of the network that represents the correct parse; this makes the network less dependent on the particular choice of weights and thresholds.

The next logical step in this research is the addition of a semantic component in our scheme, to extend the disambiguation capability. Such a model would incorporate rules for case filling.

References

COTTRELL, G.W. (1985). *A connectionist approach to word sense disambiguation.* Doctoral dissertation, Computer Science Department, University of Rochester, Rochester, NY 14627, April 1985.

COTTRELL, G.W. and SMALL, S.L. (1983). A connectionist scheme for modelling word sense disambiguation. *Cognition and Brain Theory*, **6**(1), 1983, 89-120.

FAHLMAN, S.E.; HINTON, G.E. and SEJNOWSKI, T.J. (1983). Massively parallel architectures for AI: NETL, Thistle, and Boltzmann machines. *Proceedings of the National Conference on Artificial Intelligence*, Washington, August 1983, 109-113.

FELDMAN, J.A. and BALLARD, D.H. (1982). Connectionist models and their properties. *Cognitive Science* **6**, 1982, 205-254.

HINTON, G.E. and SEJNOWSKI, T.J. (1983a). Analyzing cooperative computation. *Proceedings of the Fifth Annual Conference of the Cognitive Science Society*, Rochester, NY, May 1983.

HINTON, G.E. and SEJNOWSKI, T.J. (1983b). Optimal perceptual inference. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Washington DC, June 1983.

KIRKPATRICK, S.; GELATT, C.D.Jr. and VECCHI, M.P. (1983). Optimization by simulated annealing. *Science*, **220**#4598, 1983, 671-680.

McCLELLAND, J.L. and RUMELHART, D.E. (1981). An interactive activation model of context effects in letter perception. Part1, An account of basic findings. *Psychological Review*, **88**, 1981, 375-407.

POLLACK, J.B. and WALTZ, D.L. (1982). Natural language processing using spreading activation and lateral inhibition. *Proceedings of the fourth Annual Conference of the Cognitive Science Society*, Ann Arbor, August 1982, 50-53.

REILLY, R.G. (1984). A connectionist model of some aspects of anaphor resolution. *Proceedings of the* 10^{th} International Conference on Computational Linguistics, Stanford, July 1984, 144-149.

SMALL, S.T.; COTTRELL, G. and SHASTRI, L. (1982). Towards Connectionist Parsing. *Proceedings of the National Conference on Artificial Intelligence*, Pittsburgh, August 1982, 247-250.

SELMAN, B. (1985). Rule-Based Processing in a Connectionist System for Natural Language Understanding. Technical Report CSRI-168, Computer Systems Research Group, University of Toronto, April 1985.

WALTZ, D.L. and POLLACK, J.B. (1984). Phenomenologically plausible parsing. *Proceedings of the National Conference on Artificial Intelligence*, Austin, Texas, U.S.A., August 1984, 335-339.

WINOGRAD, T. (1983). *Language as a cognitive process.*, Reading, MA: Addison-Wesley Publishing Company, 1983.

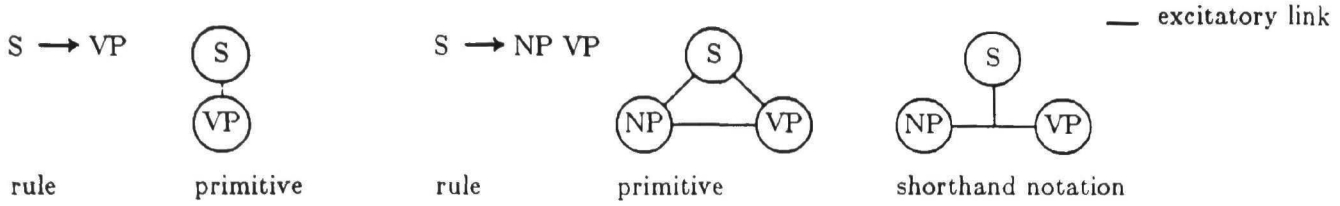

Figure 1 Two examples of connectionist primitives and their associated grammar rules.

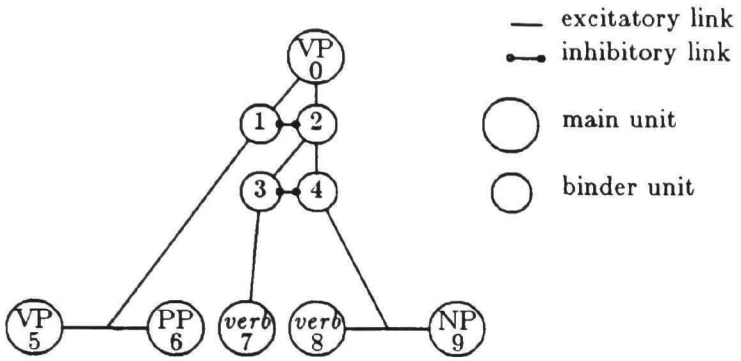

Figure 2 An example of the use of binder units (units 1, 2, 3 and 4).

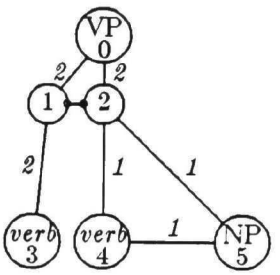

Figure 3 Some excitatory links in a typical configuration in the parsing network; their weights are given alongside the links.

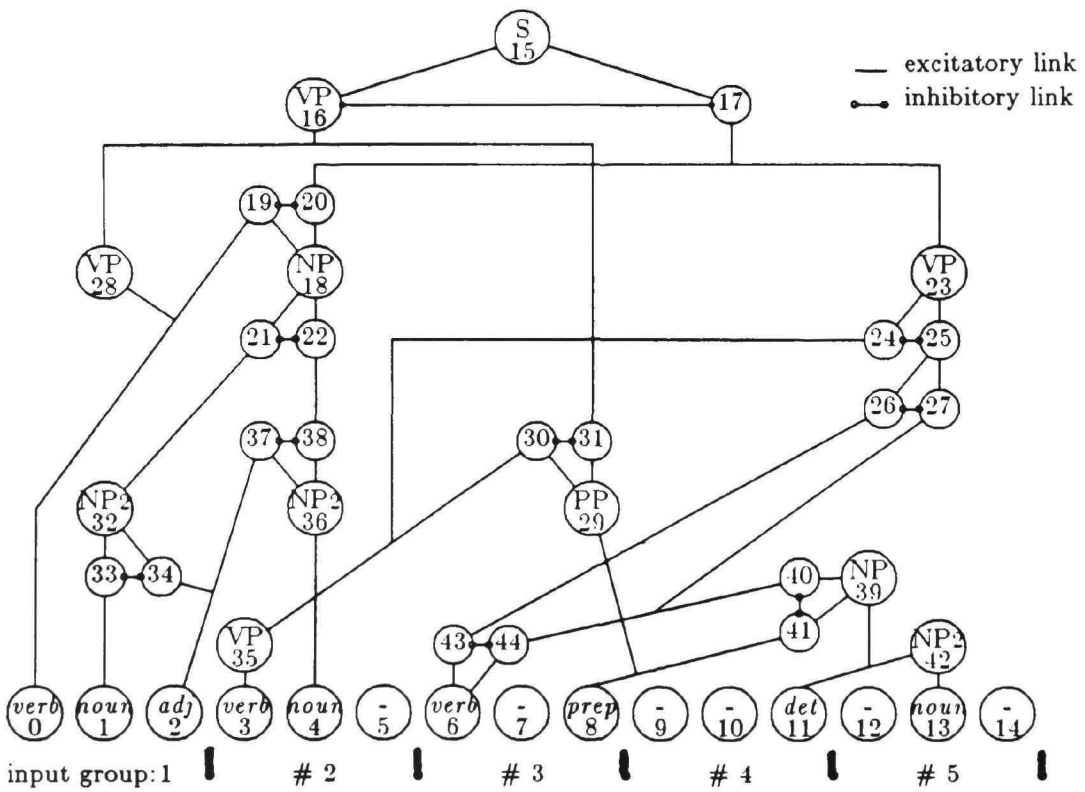

Figure 4 An example parsing network based on the grammar rules given in (6). Input group #1 consists of units 0, 1, and 2; group #2 consists of units 3, 4, and 5, and so forth.

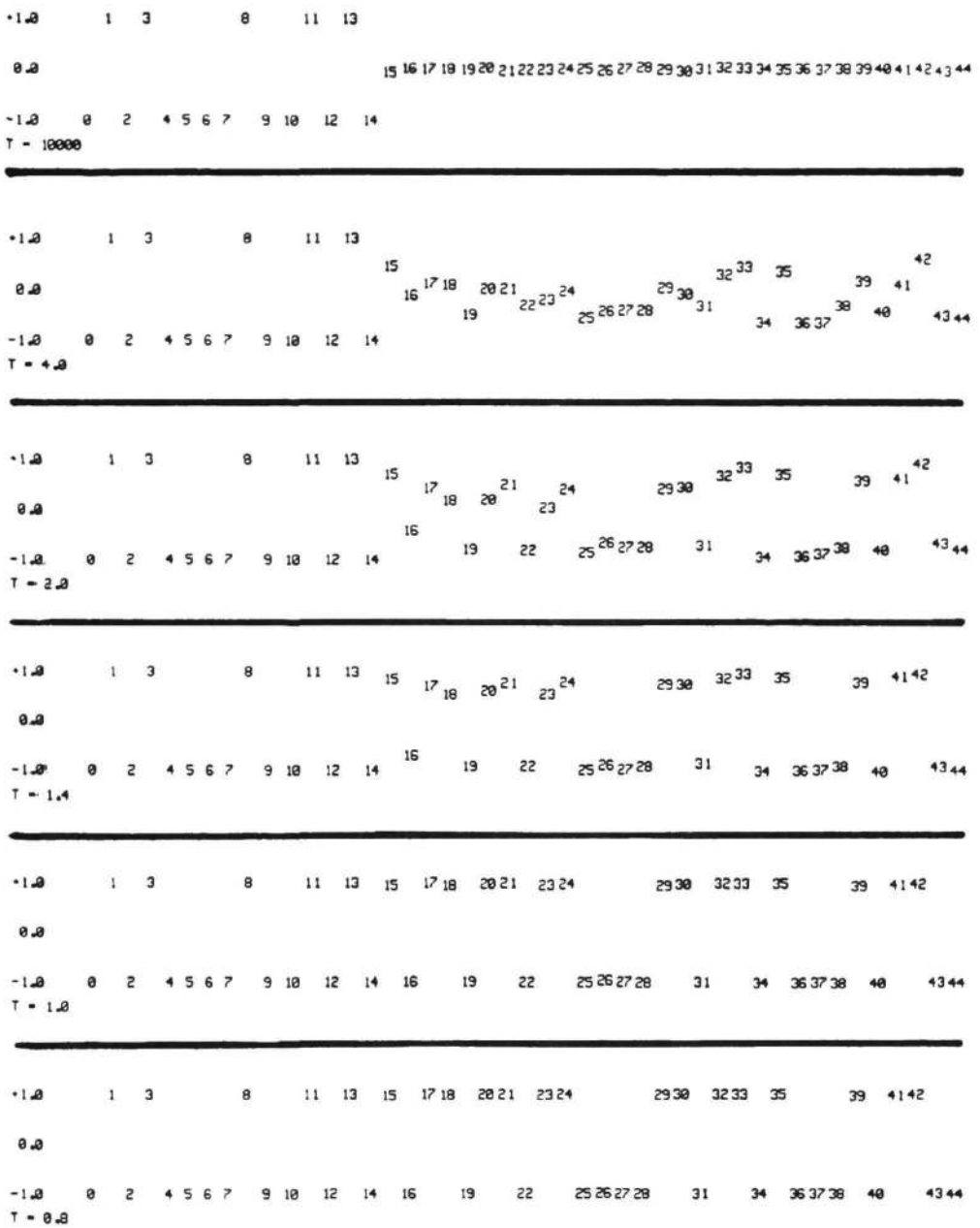

Figure 5 Average output values of computing units during simulated annealing. During the simulations the outputs of units 0 to 14 were fixed to represent the input sentence.

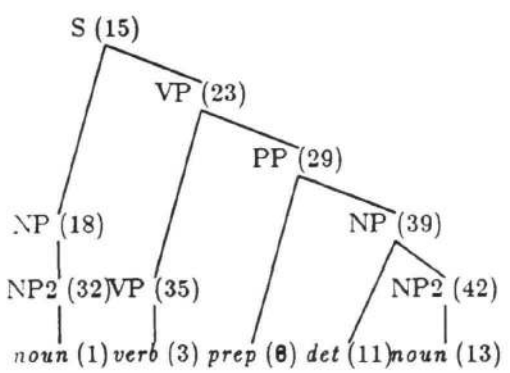

Figure 6 The parse tree of sentence (7). The numbers between parentheses are the numbers of the corresponding computing units in the parsing network.