presented results, it can be expected that the algorithm will successfully handle octrees of real objects if their shape is relatively well preserved. For this reason, both a good octree construction method and an appropriate resolution influenced by an object's complexity have to be chosen carefully. Note that at higher resolutions, the threshold values discussed here can be reduced, but since the values to which the thresholds are applied decrease only for symmetric objects, the values given here can be retained without a significant loss of the algorithm's sensitivity.

V. Conclusion

An algorithm for identifying symmetry of an arbitrary 3-D object represented by an octree is presented and implemented. It is applicable to any object regardless of its shape, and in general, it works solely with an object's (possibly noisy) octree representation without any need for preprocessing or additional data (which may be necessary in some special cases). A wide range of symmetry types represented by groups of proper and improper rotations can be identified and evaluated by means of a continuous measure of symmetry called the *symmetry degree*. To the best of our knowledge, this is the first practical algorithm for symmetry identification of 3-D objects that can successfully handle noisy input data. The algorithm has been verified by selected examples using synthetic objects.

Identification of spherical symmetry should be realized in a more efficient manner. To do this, the positioning of objects using higher order 3-D moments is presently under investigation. Recently, some complex 3-D moment forms were evaluated [13], but almost all of them vanish in such a case. Thus, the general positioning scheme remains to be found. Once this is done, the procedure described here can be used for symmetry evaluation using the octree only (without additional data).

Symmetry is a very important feature of objects, and we believe that its automatic identification will find numerous applications in computer vision, intelligent CAD systems, and other related fields.

Acknowledgment

The authors wish to thank anonymous reviewers for their valuable comments, K. Sugihara of Tokyo University and J. Bigun of Ecole Polytechnique Federale de Lausanne for kindly providing related literature, and M. Vevera for his help in editing.

References

[1] Y. S. Abu-Mostafa and D. Psaltis, "Image normalization by complex moments," *IEEE Trans. Patt. Anal. Machine Intel.*, vol. PAMI-7, pp. 46–55, 1985.

[2] N. Alexandridis and A. Klinger, "Picture decomposition, tree data structures, and identifying directional symmetries as node combination," *Comput. Graphics Image Processing*, vol. 8, pp. 43–77, 1978.

[3] M. D. Atkinson, "An optimal algorithm for geometrical congruence," *J. Algorithms*, vol. 8, pp. 159–172, 1987.

[4] J. Bigun, "Local symmetry features in image processing," Ph.D. thesis, Linkoeping Univ., Linkoeping, Sweden, 1988, no. 179 (ISBN 91-7870-334-4).

[5] ——, "A structure feature for some image processing applications based on spiral functions," *Comput. Vision Graphics Image Processing*, vol. 51, pp. 166–194, 1990.

[6] C. H. Chien and J. K. Aggarwal, "Identification of three dimensional objects from multiple views using quadtrees/octrees," *Comput. Vision Graphics Image Processing*, vol. 36, pp. 256–273, 1986.

[7] P. Eades, "Symmetry finding algorithms," in *Computational Morphology* (G. T. Toussaint, Ed.). Amsterdam: North-Holland, 1988, pp. 41–51.

[8] T. G. Evans, "A program for the solution of a class of geometric-analogy intelligence-test questions," in *Semantic Information Processing* (M. Minsky, Ed.). Cambridge, MA: MIT Press, 1968, pp. 271–353.

[9] S. A. Friedberg, "Finding axes of skewed symmetry," *Comput. Vision Graphics Image Processing*, vol. 34, pp. 138–155, 1986.

[10] J. Gips, "A syntax-directed program that performs a three-dimensional task," *Patt. Recogn.*, vol. 6, pp. 189–199, 1974.

[11] R. C. Gonzalez and P. Wintz, *Digital Picture Processing*. Reading, MA: Addison-Wesley, 1987.

[12] J. J. Leou and W. H. Tsai, "Automatic rotational symmetry determination for shape analysis," *Patt. Recogn.*, vol. 20, pp. 571–582, 1987.

[13] C. H. Lo and H. S. Don, "3-D moment forms: their construction and application to object identification and positioning," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 11, pp. 1053–1064, 1989.

[14] G. Marola, "On the detection of the axes of symmetry of symmetric and almost symmetric planar images," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 11, pp. 104–108, 1989.

[15] P. Minovic, S. Ishikawa, and K. Kato, "Three-dimensional symmetry identification," *Memoirs Kyushu Inst. Tech. (Eng.)*, no. 21, pp. 1–38, 1992.

[16] H. Noborio, S. Fukuda, and S. Arimoto, "Construction of the octree approximating three dimensional objects by using multiple views," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 10, pp. 769–782, 1988.

[17] H. Samet, *Application of Spatial Data Structures*. Reading, MA: Addison-Wesley, 1990.

[18] S. K. Srivastava and N. Ahuja, "Octree generation from object silhouettes in perspective views," *Comput. Vision Graphics Image Processing*, vol. 49, pp. 68–84, 1990.

[19] K. Sugihara, "An $n \log n$ algorithm for determining the congruity of polyhedra," *J. Comput., Syst. Sci.*, vol. 29, pp. 36–47, 1984.

[20] K. R. Symon, *Mechanics*. Reading, MA: Addison-Wesley, 1965.

[21] J. Weng and N. Ahuja, "Octrees of objects in arbitrary motion," *Comput. Vision Graphics Image Processing*, vol. 39, pp. 167–185, 1987.

[22] H. Weyl, *Symmetry*. Princeton, NJ: Princeton Univ. Press, 1952.

[23] J. D. Wolter, T. C. Woo, and R. A. Volz, "Optimal algorithms for symmetry detection in two and three dimensions," *Visual Comput.*, vol. 1, pp. 37–48, 1985.

[24] M. Yau, "Generating quadtrees of cross sections from octrees," *Comput. Vision Graphics Image Processing*, vol. 27, pp. 211–238, 1984.

[25] E. Yodogawa, "Symmetropy, an entropy-like measure of visual symmetry," *Perception Psychophys.*, vol. 32, no. 3, pp. 230–240, 1982.

[26] L. Zusne, "Measures of symmetry," *Perception Psychophys.*, vol. 9, no. 3B, pp. 363–366, 1971.

Solving Satisfiability via Boltzmann Machines

A. d'Anjou, M. Graña, F. J. Torrealdea, and M. C. Hernandez

***Abstract*—Boltzmann machines (BM's) are proposed as a computational model for the solution of the satisfiability (SAT) problem in the propositional calculus setting. Conditions that guarantee consensus function maxima for configurations of the BM associated with solutions to the satisfaction problem are given. Experimental results that show a linear behavior of BM's solving the satisfiability problem are presented and discussed.**

***Index Terms*—Boltzmann machines, satisfiability, simulated annealing.**

Manuscript received July 30, 1990; revised October 25, 1991. This work was supported by grants UPV 140.226-E161/90 from the UPV/EHU and OF851 of the Department Economia, Excma. Diputación foral de Guipuzeoa. Recommended for acceptance by Associate Editor R. DeMori.

The authors are with Facultad Informática UPV/EHU, San Sebastián, Spain.

IEEE Log Number 9208437.

I. Introduction

The problem of the satisfiability (SAT) of an expression in the propositional calculus, which is usually given in conjunctive normal form, was one of the first problems to be characterized as NP complete in its general statement. Linear complexity algorithms have been proposed [9] for HORN-SAT, which is a special class of SAT problems in which the clauses of the expression are propositional Horn clauses. However, the general exact resolution algorithms (either the classical propositional version of the Davis–Putnam algorithm [7] or the more recent improvements of it by Purdom [20] and Monien {it et al.} [18]) are implicit enumeration algorithms. In the recent literature, relaxation schemes have been proposed in [10] to map instances of SAT into instances of HORN-SAT and to use the relaxed subproblems as a pruning device. Another approach that provides an improved, but still exponential nature, tester for nonclausal propositional expressions is studied in [11]. The work of Pinkas [19] deserves special mention because it gives formal foundations for the mapping of SAT problems into energy functions that can be realized as recurrent neural networks.

The problem of finding the maximum subset of clauses that can be satisfied simultaneously (MAX-SAT) can be viewed as a generalization of SAT. For the approximate resolution of MAX-SAT, some optimization methods have been proposed [12], [13], [17]. The present work can be framed under these trends to solve MAX-SAT. Approximate solutions of MAX-SAT are used to answer SAT. Negative answers to SAT given on these grounds have some degree of uncertainty due to nonzero probability of suboptimal resolution of MAX-SAT.

Boltzmann machines (BM's), which were originally introduced in [6], have been proposed in [1]–[3], [5], and [22] as a model for the massively parallel implementation of the simulated annealing algorithm [15], [16]. Although it requires large computational resources [14], simulated annealing has proven to be a powerful algorithm for the approximate solution of combinatorial optimization problems. The intended application of a BM [6] was as associative memories through a learning process. However, later works [4] have been addressed to the design of a BM for the approximate solution of combinatorial optimization problems such as the TSP.

A BM can be viewed as a special case of the more general connectionist models. The computing elements are interconnected logical units (with binary local states). A strength is associated with each connection, which gives a quantitative measure of the desirability of the simultaneous activation of both units. A global state or configuration is a pattern of states of the logical units. A consensus function assigns to each configuration a real value that is the global quantitative measure of its goodness. The individual state of each unit is determined through a stochastic function of the states of its neighbors and the strengths of their interconnections. When solving optimization problems, consensus maximization (through simulated annealing) provides the desired optimum. The potential for massive parallelism in BM's arises from the locality property of the computation of the consensus function increments that guide the search for its maximum.

As stated previously, we consider SAT embedded in MAX-SAT. Solutions to MAX-SAT that equal the whole set of clauses allow positive answers to SAT; the failure to find them provides speculative negative answers to SAT. Bearing this idea in mind, we have concentrated our efforts in the definition of a class of BM fitted to the MAX-SAT problem. Final configurations of the BM formulated for a given set of clauses map into truth assignments that given partial or total satisfactions of it. The proposed formulation allows progressive construction, and changes in the set of clauses can be easily mapped into the BM.

Although it is theoretically possible to guarantee the proper convergence of the BM, the approximate character of any finite realization introduces a nonzero probability of error that would occur when a set of clauses is declared as unsatisfiable when they are, in fact, satisfiable. In addition, the stochastic nature of a BM makes it quite unfit for applications where an exhaustive enumeration of truth assignments that satisfy the set of clauses is required. Nevertheless, this approach allows the solution of the SAT problem in its more general form and gives a model for its massively parallel computation. The most appealing result of our work is that BM's seem to behave linearly on the number of propositions involved and are quite unaffected by other complexity factors: the number of clauses and their structure (whether they are Horn clauses, their size, etc.). This behavior can be explained partially by the fact that some of the parameters that determine the computational complexity of the annealing algorithm are defined in this work in terms of the number of propositions involved.

The remainder of this section introduces the notation employed in the statement of the satisfiability problem and BM's. Section II presents the construction of a BM to solve an instance of SAT, which are the conditions that guarantee that maximum consensus configurations correspond to solutions and our approach to the realization of these conditions. Section III presents the experimental framework. Section IV presents the discussion of the results gathered from the experiments. Finally, in Section V, we give our conclusion and some discussion over the material of previous sections.

A. General Statement of the Satisfiability Problem

Let $\mathcal{E}$ be an expression of the propositional calculus in conjunctive normal form:

$$\mathcal{E} = \wedge_{j=1..NC} C_j$$

where NC is the number of clauses, $C_j = \vee_{i=1..n_j} \mathcal{L}_i$ is a clause in disjunctive form, n_j is the number of literals in clause C_j, a literal $\mathcal{L}_i$ can be either a proposition $\mathcal{P}_i \ \ i \in \{1..NP\}$ or its negation, and each proposition is a Boolean variable.

The satisfiability problem can be stated as the search for a truth assignment $\mathcal{A} : \{\mathcal{P}_i \ \ i \in \{1..NP\}\} \rightarrow \mathbb{B}$ such that the evaluation of $\mathcal{E}$ under $\mathcal{A}$ is true.

The maximum satisfiability problem can be stated as the search for the maximum $\mathcal{E}^* \subseteq \mathcal{E}$ such that $\mathcal{E}^*$ is satisfiable. Obviously when $\mathcal{E}^* = \mathcal{E}$, the answer to SAT posed on $\mathcal{E}$ is yes.

B. Definitions for BM's

A BM with N logical units can be represented as an undirected graph $G = (U, C)$, where the vertex set $U = \{u_0, \ldots, u_{N-1}\}$ is the set of units, and the edge set $C \subseteq U \times U$ denotes the set of connections between the units. A connection $(u_i, u_j) \in C$ connects the units u_i and u_j. C includes connections of units with themselves.

The state of each unit can be either 0 or 1. A configuration k of the BM is univocally determined by the states of its units. The state of unit u_i in configuration k is denoted by $k(u_i)$. The configuration space R denotes the set of all possible configurations ($|R| = 2^N$).

A connection (u_i, u_j) is activated in the configuration k if $k(u_i)k)(u_j) = 1$. In a connection (u_i, u_j), a strength $s(u_i, u_j)$ is associated with a real number that is interpreted as a quantitative measure for the desirability that the connection is activated.

The consensus function $C : R \rightarrow \mathbb{R}$ is a measure of the global goodness of configuration k and is defined as follows:

$$C(k) = \Sigma_C s(u_i, u_j) k(u_i) k(u_j).$$

The set $R_k \subset R$ is the neighborhood of configuration k and is defined as the set of configurations that differ from configuration k

only in the state of one unit. Let $k' \in R_k$ be a neighbor of k, and the states of the units k' are given by

$$k'(u_j) = \begin{cases} k(u_j) & \text{if } j \neq i \\ 1 - k(u_j) & \text{if } j = i. \end{cases}$$

A very interesting property of BM's is the locality of the computation of consensus variations between neighboring configurations. Let C_i be the set of connections affecting unit u_i, excluding the bias (u_i, u_i). The consensus variation between two neighboring configurations can be computed as follows:

$$\begin{aligned}\Delta C(kk') &= C(k') - C(k) \\ &= (1 - 2k(u_i)) \\ &\quad \cdot [\Sigma_{C_i} s(u_i, u_j) k(u_j) + s(u_i, u_i)].\end{aligned}$$

This locality in the computation of consensus variations is at the heart of the proposition of a BM as a massively parallel computational model.

The objective of the BM is to reach a configuration with global maximum consensus. In general, this is achieved through a simulated annealing strategy. In its sequential (uniprocessor) realization, neighboring configurations are generated by random selection of the unit whose state is to be changed. The probability of accepting this generated configuration as the new configuration of the machine is defined as

$$\boldsymbol{A}_{kk'}(\boldsymbol{t}) = [1 + e^{-\Delta C/\boldsymbol{t}}]^{-1}$$

where ΔC is the consensus increment $\Delta C(kk')$ whose expression is given above, and $\boldsymbol{t}$ is the control parameter, which is usually referred as temperature. In concurrent (multiprocessor) realizations, units are clustered in groups that proceed in parallel with their local annealing processes. In the limit, each unit changes its state concurrently and asynchronously (massive parallelism), following a local annealing process based on uncertain information about the states of the units connected to it. To our knowledge, the precise effects of concurrent computation on BM accuracy remain an open research topic.

The simulated annealing algorithm is mathematically conceptualized as a sequence of Markov chains, where each Markov chain is a sequence of trials to change the configuration of the machine at a constant temperature value. Any finite realization of the simulated annealing algorithm thus requires the specification of a cooling schedule consisting of 1) the start value $\boldsymbol{t}_0$ of the temperature, 2) the decrement function of the temperature, 3) the length of each Markov chain, and 4) the stop criterion for the algorithm. In Section III, we specify the cooling schedules used in our experimental work.

II. BM's for SAT

In this section, we present the definition of a class of BM fitted for the resolution of MAX-SAT posed on a given set of clauses. Solutions to MAX-SAT obtained via these BM's will be used in the experiments reported later to give answers to the SAT problem. In Section II-A, we describe the construction of the BM for a given propositional expression. We also define the notion of consistent configuration. In Section II-B, we give the conditions with which the forces associated; and what the connections must accomplish in order to guarantee the theoretical convergence to configurations that correspond to truth assignments giving maximum satisfaction of the propositional expression. In Section II-C, we describe our approach to the realization of the aforementioned conditions.

A. Construction of the BM

Let $\boldsymbol{BM}_{\mathcal{E}}$ be the BM that will serve to approach the solution to MAX-SAT posed on the propositional expression $\mathcal{E}$. The set of units of $\boldsymbol{BM}_{\mathcal{E}}$ is

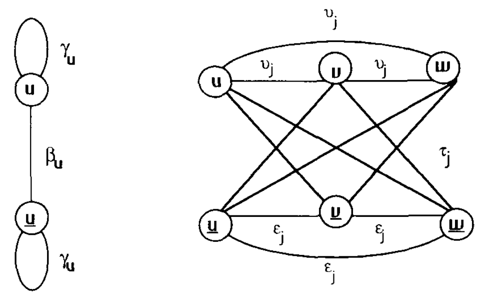

Fig. 1. Graphical representation of units, connections, and their strengths.

$$U_{\mathcal{E}} = \{0 \ldots 2\text{NP} - 1\}.$$

Unit $u \in U_{\mathcal{E}}$ represents a literal that can either be the proposition $\mathcal{P}_{u+1}$ if $u <$NP or the negation of proposition $\mathcal{P}_{u-\text{NP}+1}$ if $u \geq$ NP. For convenience, let us define

$$\underline{u} = (u + \text{NP}) \bmod 2\text{NP}.$$

This function computes the unit that represents the literal that is the negation of the literal represented by unit u. To ease the notation, we will assume that clause C_j is described by the set of units that represent the literals that appear in it, and we denote this set by $C_j \subset U_{\mathcal{E}}$, and its cardinality $|C_j|$ will be the number n_j.

A configuration $k \in R$ of the BM is defined to be *consistent* if and only if the following holds in k:

$$k(u) = 1 - k(\underline{u}) \quad u \in \{0 \ldots \text{NP} - 1\}.$$

In words, for a configuration k to be consistent, any pair of units u and $\underline{u}$ (a literal and its negation) must be in different states. Consistency defines a partition of the configuration space:

$$R = R^{+} U R^{-}$$

where R^{+} denotes the set of consistent configurations and R^{-} that of inconsistent ones. Consistent configurations can be associated with a truth value assignment $\mathcal{A}_k$ on the propositions $\mathcal{P}_i$ $i \in \{1 \ldots \text{NP}\}$ as follows:

$$\mathcal{A}_k(\mathcal{P}_i) = \begin{cases} \boldsymbol{T} & \text{if } k(i-1) = 1 \text{ and } k(\underline{i-1}) = 0 \\ \boldsymbol{F} & \text{if } k(i-1) = 0 \text{ and } k(\underline{i-1}) = 1. \end{cases}$$

The set of connections $C_{\mathcal{E}}$ is partitioned into three sets:

$C^{b} = \{(u, u) | u \in U_{\mathcal{E}}\}$ are the bias connections.

$C^{e} = \{(u, \underline{u}) | u \in \{0 \ldots \text{NP} - 1\}\}$ are the exclusion connections between units representing a proposition and its negation.

$C^{c} = \{(u, v) | u, v \in U_{\mathcal{E}} \wedge u \neq \underline{v}\}$ are the connections used to represent the clauses.

Connections in C^{b} and C^{e} will play a definite role in the dynamic search for consistent configurations. For this reason, we detach their associated forces and denote them as $\gamma_u = s(u, u)$ and $\beta_u = s(u, \underline{u})$. Fig. 1(a) shows the bias and exclusion connections and forces between a pair of complementary units.

The partial forces that model each clause in the BM are as follows: for each pair of units representing literals in the clause C_j, a force ϵ_j between the negations of the elements of the pair, a force τ_j between each element of the pair and the negation of the other one, and a force v_j between the elements of the pair.

$$\begin{aligned}&n_j > 1 \;\; \forall u, v \in C_j \;\; s_j(\underline{u}, \underline{v}) = \varepsilon_j, \\ &\quad s_j(\underline{u}, v) = s_j(u, \underline{v}) = \tau_j, s_j(u, v) = v_j. \qquad (1)\end{aligned}$$

This construction applies to clauses with two or more literals ($n_j \geq 2$). Fig. 1(b) shows the partial forces that model a clause $C_j = \{u, v, w\} \subset U_\mathcal{E}$. The final construction of forces of the BM is obtained by adding up the partial forces due to each clause:

$$s(u, v) = \Sigma_{j=1 \ldots NC} s_j(u, v) \quad (u, v) \in C''.$$

This definition of the forces has two advantages: the first allows progressive construction and easy addition or subtraction of a clause. The second allows the expression of the consensus function in terms of a satisfaction function of each clause, which in turn allows us to reason in terms of clauses when analyzing the consensus variations in the next section. (Reference [8] contains the details.)

From the above construction of the connections and forces, we can state that the generic expression of the consensus function can be decomposed into three addends: the summation of the bias forces, the summation of the excluding forces, and the summation of the remainder forces. It must be noted that for all consistent configurations, the second addend will always be zero.

B. Conditions on the Connection Strengths

As stated in Section I-B, the dynamic behavior of BM's consists of the search for maximum consensus configurations. We want final configurations of the BM's, which were built up following the description in Section II-A, to be consistent and to satisfy as many clauses as possible, all of them in the case of a satisfiable set of clauses.

In order to guarantee that any final configuration is a consistent one, we must assure that consistent configurations correspond to local maxima of the consensus function. From now on, we will denote $u \in U_\mathcal{E}$ the unit whose state change gives way to the neighboring configuration. It must be noted that each neighboring configuration of a consistent one is inconsistent ($\forall k \in R^+, k^u \in R_k \Rightarrow k^u \in R^-$). Formally, we try to guarantee the following:

$$\forall k \in R^+ \quad k^u \in R_k \quad \Delta C(kk^u) = C(k^u) - C(k) < 0. \tag{2}$$

We find two distinguishable cases of transition to inconsistent neighbors from a consistent one: Either unit u becomes inactive or active in the new configuration.

In the first case, the transition from consistency to inconsistency produces the disappearance of some forces: the bias γ_u of u and all the forces associated with connections from u to any other active unit. Again, (2) implies that γ_u must compensate the summation of all the other forces incident on unit u that will disappear when it becomes inactive. Formally, for a given configuration k, this condition reads

$$\gamma_u > -\Sigma_{C^u} s(u, v)k(v). \tag{3}$$

In order to guarantee (2), irrespective of particular configurations and units, we define a common value γ for the bias term of all the units as follows:

$$\gamma > -S_1 \tag{4}$$

where $S_1 = \min \{\Sigma_{C^u} s(u, v) \ \ \forall u \in U_\mathcal{E}\}$.

In the second case, the transition from the consistent configuration to the inconsistent one produces the activation of new forces: a force β_u between units u and $\underline{u}$, the bias γ of u (using (4)), and all the forces associated with connections between u and any other active unit. For the fulfillment of (2), β_u must not be compensated by the value of the addition of the remainder new forces established by the activation of unit u. Let $C^u = \{(u, v) | v \in U \wedge v \neq u \wedge v \neq \underline{u}\}$ for a given unit u. Formally, this condition on β_u can be written as follows:

$$\beta_u < -[\gamma + \Sigma_{C^u} s(u, v)k(v)]. \tag{5}$$

Again, in order to guarantee (2) irrespective of particular configurations and the chosen unit u, we define β for all the units as

$$\beta < -[\gamma + S_2] \tag{6}$$

where $S_2 = \max \{\Sigma_{C^u} s(u, v) \ \ \forall u \in U_\mathcal{E}\}$.

Values of γ and β fulfilling (4) and (6) guarantee that the BM converges to consistent configurations. Condition (4) is trivial if $S_1 > 0$. In general, γ will be a positive value, whereas β will be negative.

To study the degree of satisfaction of final configurations, let us consider a new partition of the consistent configuration space:

$$R^+ = R^0 U R^1 U \ldots U R^{NC}$$

where $R^n = \{k | \mathcal{A}_k \text{ satisfies } n \text{ clauses}\}$.

To assure that final configurations satisfy as many clauses as possible, we must define the forces that represent the clauses in such a way that the consensus function increases monotonically with the increase of the index in this new partition:

$$k \in R^n \quad k' \in R^{n-1} \quad \Delta C(kk') = C(k') - C(k) < 0.$$

To analyze the form of the consensus variation between those regions of the configuration space, we developed an expression of it in terms of the variations of the satisfaction function of each clause:

$$k \in R^+ \quad C(k) = \text{NP}\gamma + \Sigma_{j=1 \ldots NC} \Phi(C_j, k).$$

The satisfaction function $\Phi(C_j, k)$ computes the contribution to the consensus function due to the degree of satisfaction of the clause C_j in the configuration k. The way in which the forces of the BM are constructed (this is described in (1)) allows the computation of this function. Formally, the function reads

$$\Phi(C_j, k) = \binom{\pi(j, k)}{2} \upsilon_j + \pi(j, k)\underline{\pi}(j, k)\tau_j + \binom{\underline{\pi}(j, k)}{2} \varepsilon_j \tag{7}$$

where $\pi(j, k)$ denotes the number of units in C_j that are activated in configuration k, and $\underline{\pi}(j, k) = n_j - \pi(j, k)$. The clause C_j would be satisfied under the truth assignment $\mathcal{A}_k$ associated with the configuration k if $\pi(j, k) \geq 1$.

The variation of the consensus function between any two consistent configurations can be expressed in terms of the consensus variation between near-neighboring configurations, which only differ in the value assigned to one proposition by considering a chain of such near-neighboring configurations. In its turn, the consensus variation between near-neighboring configurations can be expressed in terms of the variation of the satisfaction function of the individual clauses.

Let k and k' be near-neighboring consistent configurations, and let us assume that $\pi(j, k') = \pi(j, k) + 1$. This assumption does not affect the generality of the statements to follow, and it means that clause C_j has a greater degree of satisfaction in configuration k'; it also implies that $\pi(j, k) < n_j$. From (7), the variation of the satisfaction function of clause C_j between these configurations is of the form:

$$\Delta\Phi(C_j, kk') = \pi(j, k)[\upsilon_j - 2\tau_j + \varepsilon_j] + (n_j - 1)[\tau_j - \varepsilon_j]. \tag{8}$$

In order for the BM to be able to discriminate between the regions of the partition (defined previously) over the consistent configuration space, the following conditions must be met by the satisfaction functions of the clauses (see [8] for details):

The variation of consensus introduced by the satisfaction of a clause must be a positive quantity (let us say α); therefore, the increment of the satisfaction function must be greater than or equal to α when the number of active units in C_j (the clause) goes from 0 to 1. Formally

$$\Delta\Phi(C_j, kk') > \alpha \text{ when } \pi(j, k) = 0. \tag{9}$$

The variation of consensus introduced by a varying degree of satisfaction of a set of clauses cannot accumulate up to the point of compensating and covering a satisfaction or dissatisfaction. Therefore, the absolute value of the increment of the satisfaction function, when the number of active units goes from one onwards, must be as small as required in order to guarantee that. Formally

$$|\Delta\Phi(C_j, kk')| < \alpha/\mathrm{NC}n_m \text{ when } \pi(j, k) \geq 1 \tag{10}$$

where $n_m = \max\{n_j \;\; j = 1 \ldots \mathrm{NC}\}$.

Conditions (9) and (10) cannot be met by a function such as the satisfaction function of the clauses whose increments are produced in a linear fashion. In the next section, we describe how we have approached the realization of these conditions from theoretical and implementation points of view.

C. Realization of the Conditions

To overcome the limitations in discrimination imposed by the linearity of the consensus increments, we proposed, in [8], an extension of the BM definition, including a new kind of unit whose states are deterministic functions of the states of their neighbors. These new units would represent the clauses as a whole, and the activation of their bias force would produce the desired discontinuity in the increments of the consensus function. Our deterministic units are equivalent to high-order connections proposed in [21] and to sigma-pi units in [19].

The set of units of $\boldsymbol{BM}^*_{\mathcal{E}}$ (the extended BM used to solve MAX-SAT posed on $\mathcal{E}$) is

$$U^*_{\mathcal{E}} = U_{\mathcal{E}} U U^a$$

where $U^a = \{2\mathrm{NP}, \ldots, 2\mathrm{NP}+\mathrm{NC}\}$ are the new units that represent the clauses as a whole. The set of connections is constructed as

$$C^*_{\mathcal{E}} = C_{\mathcal{E}} U C^a U \{(c, c) | c \in U^a\}$$

where $C^a = \{(c, u) | c \in U^a \wedge u \in C_c\}$ are information connections between the "clause units" and the "literal units," whose strengths are

$$s(c, u) = 0 \quad (c, u) \in C^a$$

The bias forces of the "clause units" are defined as

$$s(c, c) = \alpha \quad c \in U^a.$$

The remaining forces are constructed following (5) and (6), related to consistent configurations, and (10), on the variation of the satisfaction functions of the clauses.

The determination of the state of the "clause units" in a configuration k is as follows:

$$k(c) = 1 - \Pi C_c(1 - k(u)).$$

For the experiments reported in this paper, we decided to realize this extension through a modification of the simulated annealing algorithm. For a given set of clauses $\mathcal{E}$, the $\boldsymbol{BM}_{\mathcal{E}}$ is built up as described previously, and the computation of the acceptance probability for a transition takes into account the number of clauses being unsatisfied by the destination configuration.

More precisely, the forces ε_j, τ_j, and v_j are selected in such a way that the increment in the satisfaction function produced by the activation of a unit belonging to a clause is a fixed quantity $\theta = [\alpha/\mathrm{NC}n_m]$. These values can be obtained, from (8), using the following equations:

$$(n_j - 1)[\tau_j - \varepsilon_j] = \theta \tag{11a}$$
$$v_j - 2\tau_j + \varepsilon_j = 0. \tag{11b}$$

The consensus increment is computed as usual. For each clause unsatisfied by the destination configuration, a penalty quantity $\theta\mathrm{NC}n_m$ is subtracted from the "legal" consensus variation to compute the acceptance probability. This second operation is performed only when the destination configuration is a consistent one. This test improves the performance of the algorithm and avoids ambiguities about the interpretation of the satisfaction of the clauses in configurations that are inconsistent. This approach follows the philosophy proposed in [4] of introducing penalty functions to avoid subtours in the solutions given by the BM in the LAP formulation of the TSP problem. The results presented in the next section have been obtained using this approach.

III. Experimental Framework

Section III-A presents the experimental methodology: measures observed and the general structure of experiments. Section III-B presents the concrete BM construction used for the experiments. Section III-C presents the cooling schedules used in this work.

All the experimental work has been done on SIMULA, installed on a VAX11/750, a MICROVAX, and a VAX STATION 2000.

A. Experimental Methodology

Our experimental work tries to gain insight into two topics: 1) the performance of BM's when used to verify the satisfiability of a propositional expression and 2) their reliability. We have performed three collections of experiments:

1) Experiments addressed to the tuning of the cooling scheduling of the simulated annealing strategy to be applied to the BM's
2) experiments addressed to assess the performance
3) experiments addressed to assess the reliability.

Our view of performance tries to make an abstraction from any implementation-dependent factors. For this reason, we have not gathered any account of actual CPU times, which are strongly dependent on the language used, the available hardware, and the quality of the programming, that would make it quite difficult to replicate and verify the results presented here. Instead, we have observed the number of trials in each simulated annealing of the BM and the number of annealing processes needed to find a satisfaction. The first measure is referred to from now on as "time" and the second as "iteration." From these two measures, a global performance measure, which is the total number of trials per satisfaction or "total time," can be deduced.

The criterion to decide that a propositional expression is unsatisfiable is that a maximum number of annealing processes have been attempted and that none has resulted in a satisfaction configuration. The reliability question, which is posed once this happens, is whether the expression is really unsatisfiable or if it is satisfiable, and the cooling schedule is responsible for such a misleading response. To gather information on this matter, a rough version of the DPL algorithm was implemented to verify the negative responses to SAT.

The experimentation has consisted of the random generation of a large collection of propositional expressions and application of BM's as described above to determine their satisfiability. The set of clauses of each expression were randomly generated along three parameters: number of clauses NC, number of propositions NP, and the distribution of the number of literals per clause. We have considered several distributions with clauses of constant size (two, five, six, and seven literals per clause) and a distribution with clauses of varying sizes (between two and seven with almost the same probability). This last one is referred as uniform distribution in the figures that follow. The reason to consider this parameter explicitly was to evaluate its influence on the behavior of the BM.

Our work producing 30 propositional expressions for each combination of parameter values tried has been done. In other words, each point in the figures that follow is computed as the mean of a sample of 30 sets of clauses. This sample size was assumed as reasonable for the computation of the rough confidence estimates, which are also shown in the figures.

B. Construction of the BM

Given a propositional expression described by a set of clauses, a BM that satisfies its satisfiability is constructed as described in previous sections. The concrete values assumed for the partial forces used to compute the strengths associated with the connections remain to be specified.

In order to meet (11), for each clause C_j, the values ε_j, τ_j, and v_j used to define the partial forces $s_j(u, v)$ are computed as follows:

$$\varepsilon_j = -\theta/2(n_j - 1)$$
$$\tau_j = \theta/2(n_j - 1)$$
$$v_j = 3\theta/2(n_j - 1).$$

We have assumed that the parameter θ has a value of 10. As described in Section II-A, the strengths associated with the connections that map the clauses into the BM are computed as follows:

$$s(u, v) = \Sigma_{j=1 \ldots NC} s_j(u, v) \quad (u, v) \in C^c.$$

The values for the bias forces γ and the exclusion forces β are computed to meet (4) and (6), respectively.

C. Cooling Schedules

A very conservative cooling schedule has been used. It was set after careful experimentation, which will be discussed in Section IV-A, over the set of clauses used by Purdom [20] to illustrate his own algorithm. This set of clauses was chosen because of its manageable size. In addition, the fact that it has a unique solution gives it some hard character. The cooling schedule parameters were specified so that an annealing gives satisfaction for this instance with a probability whose value is close to 0.5. This cooling schedule has the following specifications:

1) The initial temperature value $\boldsymbol{t}_o$ is computed as the summation of the absolute values of the forces divided by the number of units.

$$\boldsymbol{t}_o = \Sigma_C \text{abs}(s(u, v))/|U|.$$

2) The decrement function of the temperature is of the form

$$\boldsymbol{t}_i = \boldsymbol{f}\boldsymbol{t}_{i-1}$$

where $\boldsymbol{f} = 0.95$ is the cooling factor.

3) The length for each Markov chain is variable; its termination is produced either when the number of accepted transitions is $|U|$ or when the number of trials in this chain is $100|U|$.
4) The annealing is stopped when a chain without accepted transitions occurs.
5) The maximum number of annealing processes attempted before declaring the set of clauses as unsatisfiable is 5.

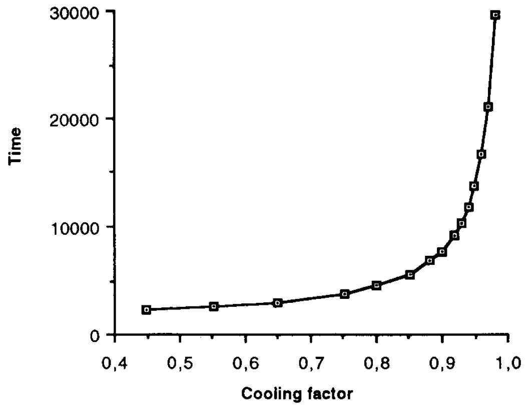

Fig. 2. Time versus cooling factor.

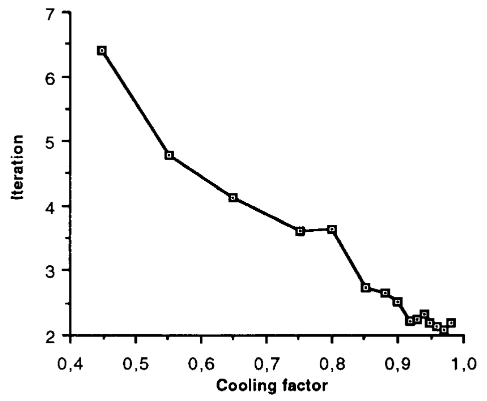

Fig. 3. Iteration versus cooling factor.

IV. Discussion of Results

In this section, we present the results from the experiments realized in the setting described above. We first discuss the tuning of the cooling schedule used along the experimentation. Then, we discuss the results gathered from the experiments addressed to assess the performance and reliability of the BM built up to give answers to the SAT problem for random instances of propositional expressions.

A. Tuning of the Cooling Schedule

To determine the appropriate value for the cooling factor $\boldsymbol{f}$, a collection of experiments with different values of it were conducted on the set of clauses given in [20]. This instance of SAT has a unique satisfaction truth assignment; therefore, the results gathered on the experimentation with it can be assumed to be valid for "hard" instances and conservative for "softer" ones, which have nonunique satisfactions.

Fig. 2 shows how the time taken for each annealing increases as the cooling factor is increased towards 1. In Fig. 3, the number of iterations needed to obtain the satisfaction of the clauses is shown at increasing values of the cooling factor. In order to provide a termination criterion for the overall process, we need a strong bound on the number of iterations required for a successful search. As show in the figure for values of the cooling factor from 0.90 onwards, the machine provides a success every two iterations. From these two figures, it must be apparent that the choice of values specified in

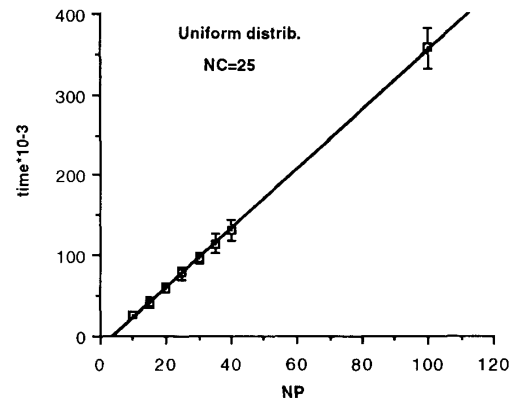

Fig. 4. Linear behavior of time versus NP.

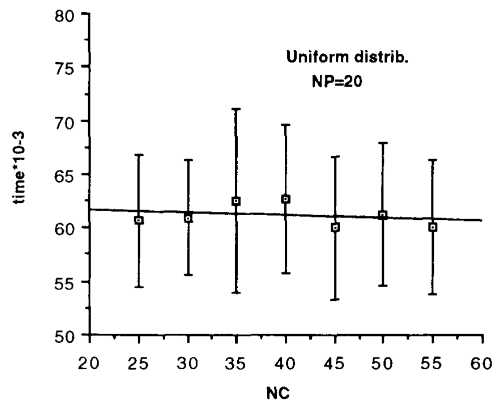

Fig. 6. Insensitivity to the number of clauses.

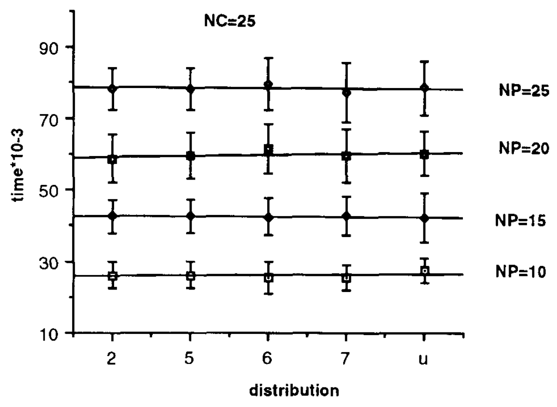

Fig. 5. Independence of time from *the* distribution of clause sizes.

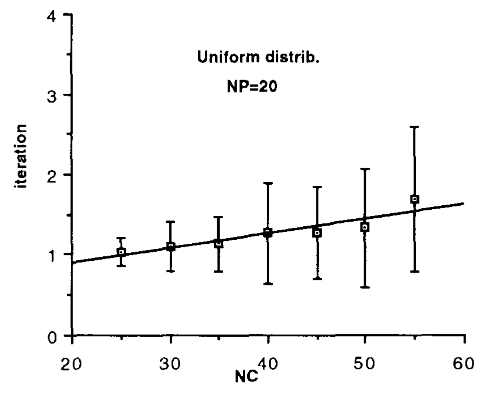

Fig. 7. Dependence of the number of iterations.

Section III-C for the cooling factor and number of iterations is quite conservative.

B. Performance Results

Fig. 4 shows the most important performance result. This figure shows the time of solution of the sets of clauses against the number of propositions involved. This performance measure has a linear behavior over the region we have tested. It must be noted that this measure does not include the time involved in the examination of the clauses needed for the correction of the consensus increment, but nevertheless, it suggests that this approach can lead to some kind of approximate "near linear" complexity solution of the SAT problem in its most general form.

Points in Fig. 4 have been obtained over sets of clauses with uniform distribution of clause size. Figs. 5–7 provide the justification for that. Fig. 5 shows the independence of the BM efficiency from the distribution of clause sizes. The distributions sampled were fixed size of 2, 5, 6, and 7 propositions per clause and the uniform (u) distribution. The figure also shows the direct dependence of the efficiency on the number of propositions (NP) involved for a given number of clauses (NC).

Fig. 6 shows the relative insensitivity of the BM efficiency to the NC involved for a given NP. As the relation NP/NC decreases, in this case by the increase of NC, it could be expected that the sets of clauses being generated would have smaller satisfaction regions so that satisfaction configurations would be more sparse and difficult to find. Fig. 6 shows that this has no effect on the time employed for each annealing, but Fig. 7 shows that this shortening of the goal region has some effect on the number of iterations required to find a satisfaction, thus increasing it. Nevertheless, the effect does not seem to be very strong.

The observation of Figs. 5–7 gives support to the idea implicit in Fig. 4: The main performance factor is the number of propositions involved. An intuitive reason for that result is that the complexity factors of the BM (number of units and connections, Markov chain length, and initial temperature) can be expressed as functions of the number of propositions, whereas the clauses are embedded in the connection strengths and mainly affect the magnitude of the consensus function.

C. Reliability Results

The experiments discussed in Section IV-B did not produce unsatisfiable cases. To systematically explore the error introduced by the approximation given by a cooling schedule, we have planned another set of experiments. We selected a number of propositions (NP=20) and a constant distribution of the clause size of two literals per clause (2-Distribution). We expected that this small clause size will increase the probability of generating unsatisfiable instances. We generated in the usual way propositional expressions with diverse numbers of clauses (NC from 25 to 50) and then applied the BM to them. Those sets of clauses declared as unsatisfiable were processed for verification by the DPL algorithm.

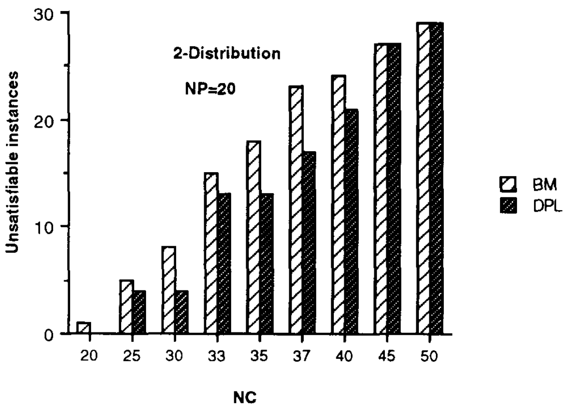

Fig. 8. Reliability experiments.

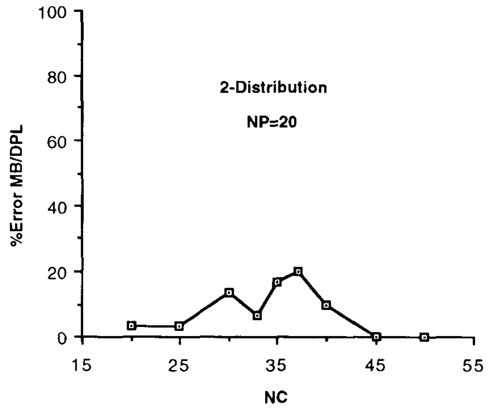

Fig. 9. Percentage of error incurred by the BM.

Fig. 8 shows the results of these experiments. The DPL bars show the number of truly unsatisfiable instances found for each size of the set of clauses. The BM bars show the number of instances declared as unsatisfiable by the BM. For the sake of clarity, we have represented in Fig. 9 the percentage of erroneous negative answers to SAT produced by the BM.

The first feature to be noticed is the increase of the probability to generate an unsatisfiable instance as the number of clauses is augmented. This confirms the intuitive idea that the ratio NP/NC is statistically related to the size of the satisfaction region of the instances being generated. The study of the error incurred by the BM shows that it reaches its maximum value when the probability of generating an unsatisfiable instance approaches 0.5. Our interpretation of this fact is as follows: It is at this point where the probability of generating satisfiable instances, whose satisfaction is a unique truth assignment, reaches a maximum value. This is, obviously, the worst context for the evaluation of any algorithm and the cause for its greater error figure.

In summary, the results show a relatively high reliability of the approach, although they also show the existence of a nonzero probability of a erroneous response, which is a distinctive feature of any approximate method.

V. Conclusions

A class of Boltzmann machines has been proposed for the resolution of the SAT problem in the propositional calculus setting. The construction of the BM for a given instance of SAT has been described. The conditions for proper convergence to solutions have been stated and their sequential realization presented. Experimental results show that the BM's proposed behave linearly on the number of propositions involved, irrespective of the number of clauses or their structure. On the other hand, the existence of a nonzero probability of error, when declaring an expression unsatisfiable, is also shown by the experiments.

Although our work has been done on sequential conventional computers, the main appeal of BM's is their massively parallel character. Further work must be addressed at the parallel realization of BM's proposed in this paper and the experimental evaluation of the performance of parallel implementations.

Acknowledgment

We wish to thank the anonymous referees for their helpful suggestions and criticism.

References

[1] E. H. L. Aarts and J. H. M. Korst, "Boltzmann machines and their applications," *LNCS*, vol. 258, pp. 34–51, 1987.

[2] ——, *Simulated Annealing and Boltzmann Machines.* New York: Wiley, 1988.

[3] ——, "Computations in massively parallel networks based on the Boltzmann machine: A review," *Parallel Comp.*, vol. 9, pp. 129–145, 1989.

[4] ——, "Boltzmann machines for travelling salesman problems," *Europ. J. Oper. Res.*, vol. 39, pp. 79–95, 1989.

[5] ——, "Boltzmann machines as a model for parallel annealing," *Algorithmica*, vol. 6, pp. 437–465, 1991.

[6] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines," *Cogn. Sci.*, vol. 9, pp. 147–169, 1985.

[7] M. Davis and H. Putnam, "A computing procedure for quantification theory, *JACM*, vol. 7, pp. 201–215, 1960.

[8] A. d'Anjou, M. Graña, F. J. Torrealdea, and M. C. Hernandez, "Máquinas de Boltzmann para la resolución del problema de la satisfiabilidad en el cálculo proposicional," *Rev. Esp. Autom. Inf.*, vol. 24, pp. 24–33, 1991.

[9] W. F. Dowling and J. H. Gallier, "Linear-time algorithms for testing the satisfiability of propositional Horn formulae," *J. Logic Prog.*, vol. 3, pp. 267–284, 1984.

[10] G. Gallo and G. Urbani, "Algorithms for testing the satisfiability of propositional formulae," *J. Logic Prog.*, vol. 7, pp. 45–61, 1989.

[11] A. van Gelder, "A satisfiability tester for nonclausal propositional calculus," *Inf. Comp.*, vol. 79, pp. 1–21, 1988.

[12] P. Hansen and B. Jaumard, "Algorithms for the maximum satisfiability problem," *Comput.*, vol. 44, pp. 279–303, 1990.

[13] D. S. Johnson, "Approximation algorithms for combinatorial problems," *J. Comp. Syst. Sci.*, vol. 9, pp. 256–278, 1974.

[14] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon, "Optimization by simulated annealing: An experimental evaluation; Part 1, Graph partitioning," *Oper. Res.*, vol. 37, no. 6, pp. 865–892, 1989.

[15] S. Kirpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Sci.*, vol. 20, pp. 671–680, 1983.

[16] P. J. M. Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications.* Dordrecht, the Netherlands: Kluwer, 1987.

[17] K. J. Lieberherr, "Algorithmic extremal problems in combinatorial optimization," *J. Algorithms*, vol. 3, pp. 225–244, 1982.

[18] B. Monien and E. Speckenmeyer, "Solving satisfiability in less than 2^n steps," *Disc. Appl. Math.*, vol. 10, pp. 287–295, 1985.

[19] G. Pinkas, "Energy minimization and the satisfiability of propositional logic" (D. S. Touretzky, J. L. Elman, T. J. Sejnowski, and G. E. Hinton, Eds.), in *Proc. 1990 Connectionist Summer Sch.*, (San Mateo, CA), 1990.

[20] P. W. Purdom, "Solving satisfiability with less searching," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, no. 4, pp. 510–513, 1984.

[21] T. J. Sejnowski, "Higher-order Boltzmann machines," *Neural Networks Comput.*, pp. 398–403, 1984.

[22] P. J. Zwietering and E. H. L. Aarts, "The convergence of parallel Boltzmann machines" (R. Eckmiller, G. Hartmann, and G. Hauske Eds.). Amsterdam: North Holland, *Parallel Processing in Neural Systems and Computers*, 1990, pp. 277–280.