Asymmetric Continuous-Time Neural Networks without Local Traps for Solving Constraint Satisfaction Problems

Botond Molnár, Mária Ercsey-Ravasz*

Faculty of Physics, Babeş-Bolyai University, Cluj-Napoca, RO-400084, Romania

Abstract

There has been a long history of using neural networks for combinatorial optimization and constraint satisfaction problems. Symmetric Hopfield networks and similar approaches use steepest descent dynamics, and they always converge to the closest local minimum of the energy landscape. For finding global minima additional parameter-sensitive techniques are used, such as classical simulated annealing or the so-called chaotic simulated annealing, which induces chaotic dynamics by addition of extra terms to the energy landscape. Here we show that *asymmetric* continuous-time neural networks can solve constraint satisfaction problems without getting trapped in non-solution attractors. We concentrate on a model solving Boolean satisfiability (*k*-SAT), which is a quintessential NP-complete problem. There is a one-to-one correspondence between the stable fixed points of the neural network and the *k*-SAT solutions and we present numerical evidence that limit cycles may also be avoided by appropriately choosing the parameters of the model. This optimal parameter region is fairly independent of the size and hardness of instances, this way parameters can be chosen independently of the properties of problems and no tuning is required during the dynamical process. The model is similar to cellular neural networks already used in CNN computers. On an analog device solving a SAT problem would take a single operation: the connection weights are determined by the *k*-SAT instance and starting from any initial condition the system searches until finding a solution. In this new approach transient chaotic behavior appears as a natural consequence of optimization hardness and not as an externally induced effect.

Citation: Molnár B, Ercsey-Ravasz M (2013) Asymmetric Continuous-Time Neural Networks without Local Traps for Solving Constraint Satisfaction Problems. PLoS ONE 8(9): e73400. doi:10.1371/journal.pone.0073400

Editor: Sergio Gómez, Universitat Rovira i Virgili, Spain

Received February 15, 2013; **Accepted** July 21, 2013; **Published** September 16, 2013

Funding: This research was supported by a Marie Curie International Incoming Fellowship FP7- PEOPLE-2011-IIF-299915. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: ercsey.ravasz@phys.ubbcluj.ro

Introduction

The most common approach in non-conventional computation is to treat dynamical systems as algorithms. Physical and biological systems are capable of achieving their functions and reaching their optimal state with incredible speed. Computer science and information technology tries to learn from nature and especially now, when CMOS technology reaches its limits at the small scale [1] (e.g. [2]), there is a hastened search for novel computational paradigms.

The use of analog dynamical systems for computation received increasing interest in the last three decades both in the theoretical and in engineering communities. Differential equations, continuous maps, and several neural network models have been employed to perform various computational tasks. In this approach, the dynamical systems are designed in a way to converge to attractors that are interpreted as the output of the computation [3–6]. Siegelmann, Orponen, Moore and others have recently provided a computational complexity theory for analog systems [5–11]. A fundamental discovery also by Siegelmann was to show, that in principle, computation beyond the Turing limit is possible. She used the strongly chaotic analog shift map as an example, and proved that it has computational power beyond the Turing machine (super-Turing) [7]. Technology has also developed devices imitating nervous system-like processing, such as the Cellular Neural/Nonlinear Network (CNN) [12,13], or analog VLSI devices [14,15]. These can solve a large variety of problems in robotics, sensory computing (vision, hearing, bionic eyeglasses) etc. The CNN is an array of analog dynamical cells performing parallel continuous-time processing, effectively solving a system of coupled ordinary differential equations (ODEs) with programmable coupling parameters. Roska and Chua proved the CNN to be at least as universal as a Turing machine [16].

Neural network models have originally been developed and investigated with the purpose of modeling brain function, however their capability of solving optimization problems has also been explored. One of the earliest works was presented by Hopfield and Tank who used neural network models to solve the traveling salesman problem [17–19]. Despite the evidently chaotic nature of brain activity and the theoretical results showing the power of chaotic analog dynamical systems [7], neural networks designed to solve combinatorial optimization problems mainly avoided chaotic dynamics, focusing on simple converging systems with Lyapunov dynamics (symmetric Hopfield networks, symmetric CNN etc.). In this approach the neural network minimizes a Lyapunov function (energy function) by converging directly to a local minimum [18,19], and this analog process is used as a basic step of the algorithm. However, for finding the global minimum classical techniques typical in digital computing are required (such as simulated annealing etc.) [8,9,20–22], and the algorithm becomes quite estranged from the original purpose of analog computing.

The need for more complex dynamics has been realized and some chaotic neural network models solving optimization problems were presented by Chen and Aihara [23,24]. These are also called as chaotic simulated annealing methods. In this approach usually the discrete-time symmetric Hopfield network is used and local traps are avoided by introducing a deterministic chaotic dynamics with a bifurcation parameter that is gradually decreased during the annealing process [23]. This method has been further improved in different ways [25–27], however the need for careful tuning of parameters has not been eliminated and direct correspondence between global optimum and the final output has not been achieved.

In our recent papers [28,29] we have shown that optimization hardness is strongly interrelated with chaotic/turbulent dynamics, implying that designing analog dynamical systems with an output that corresponds directly to the solutions of a hard problem (global minima of the energy) will necessarily show transiently chaotic behavior. Here we show that *asymmetric* continuous-time neural networks can be designed to solve hard problems simply due to their structure, without requiring a step-by-step algorithm similar to those used in digital computers. In this case transient chaotic behavior appears as a natural consequence of optimization hardness and not as an artificially added tool.

At a recent conference [30] we presented a continuous-time asymmetric neural network (CTANN) model designed to solve Boolean satisfiability (*k*-SAT) (description of the model provided below). *k*-SAT is one of the most studied constraint satisfaction problems lying at the basis of many decision, scheduling, error-correction and bio-computational applications. Our model can be transformed to solve a large variety of constraint satisfaction problems, because *k*-SAT is NP-complete, meaning that every problem in NP can be transformed into this form in polynomial time (as function of the system size) [31,32]. The NP class contains the set of optimization problems whose solutions (once given) are easily checked to satisfy the constraints, however, finding those solutions in case of hardest problems takes exponentially long search-times. For details on the computational complexity of NP-complete problems see [32].

Here we explore in details the properties of this CTANN model. We show how we can achieve a one-to-one correspondence between the *k*-SAT solutions and the stable fixed points of our CTANN. Simulations on 3-SAT, 4-SAT and 5-SAT problems show that the two important parameters do not need careful tuning during the computational process. For a given k their optimal values are fairly independent of the size and other properties of the system. We even find a common area when comparing the optimal regions for different values of k. This way non-solution traps (such as limit cycles) can be avoided and after a transiently chaotic phase the system converges to a solution.

Results

This section is organized as follows. First we briefly introduce *k*-SAT and summarize previous analog approaches including the continuous-time dynamical system introduced in [28]. Next we present our CTANN model discussing its key mathematical properties and finally we present numerical evidence on the effectiveness of the model.

Boolean satisfiability problems

In k -SAT there are given N Boolean variables, $x_i \in \{0,1\}$ and a propositional formula $\mathcal{F}$, which is the conjunction (AND) of M clauses (constraints) C_m. Each clause is the disjunction (OR, denoted by $\vee$) of k variables (x_i) or their negation ($\bar{x}_i$). In 3-SAT a constraint could be for example $C_1 = x_1 \vee \bar{x}_5 \vee x_7$. The formula may be encoded as a matrix c_{mi}:

$$c_{mi} = \begin{cases} 1 & \text{if} \quad x_i \in C_m \\ -1 & \text{if} \quad \bar{x}_i \in C_m \\ 0 & \text{if} \quad x_i \notin C_m \;\; \& \;\; \bar{x}_i \notin C_m \end{cases} \tag{1}$$

where $m = 1, \ldots, M$, $i = 1, \ldots, N$. The goal is to find an assignment of the variables such that all clauses are satisfied (TRUE).

Performance of algorithms is usually tested on random *k*-SAT instances, where each clause includes a randomly selected set of k variables. Considering that each variable could be included in its normal or negated form, these k variables can form 2^k possible clauses. We always randomly choose one of these. The simplest measure to characterize hardness of random *k*-SAT formulae is the constraint density: $\alpha = M/N$. In the easy-SAT region there are few constraints/clauses (small α) and it is easy to find solutions. For too many constraints (large α) it is easy to decide that the formula is unsatisfiable (UNSAT). There is an intermediate range (hard-SAT), however, where deciding satisfiability can be very hard: the *worst-case* complexity of any known algorithm for *k*-SAT ($k \geq 3$) is exponential in N [31]. Using statistical physics methods it has been shown that changing the constraint density α the solution space goes through several phase transitions. The hardness of problems is related to these different phases and the hardest instances appear just before the satisfiability threshold [33–35].

Previous analog approaches

There have been several attempts to solve *k*-SAT by mapping the Boolean variables onto a continuous space. In [36,37] *k*-SAT is formulated as a global optimization problem in the $\mathbf{x} \in [0,1]^N$ continuous space, however it is solved with various local search and backtracking methods characteristic to digital computing (not continuous-time dynamical systems). Refs. [38–40] use Lagrange programming neural networks for *k*-SAT. They define a Lagrange function using a linear combination of the individual constraints, with coefficients serving as Lagrange multipliers. The employed ODEs are similar to a Hopfield neural network, whose trajectories however, cannot guarantee that the corresponding dynamics does not get trapped by non-solution fixed points.

In [28] we provided a new and exact mapping of Boolean satisfiability into a set of ODEs with a unique correspondence between its set of attractors and the *k*-SAT solutions. This eliminates the key weaknesses of previous attempts. The Boolean variables are mapped to the $\boldsymbol{s} \in [-1,1]^N$ continuous space. $s_i \in [-1,1]$, $i = 1, \ldots, N$, such that $s_i = -1$ if the i^{th} Boolean variable (x_i) in the SAT problem is 0 (FALSE) and $s_i = 1$ when it is 1 (TRUE). Each constraint can be formulated as a function $K_m(\boldsymbol{s}) \in [0,1]$ which is 0 if and only if the constraint is satisfied. Accordingly an energy function can be defined as $E(\boldsymbol{s}) = \sum_{m=1}^{M} K_m^2(\boldsymbol{s})$. Finding a solution to the SAT problem (if it exists) is equivalent to finding the global minima, $\boldsymbol{s}^*$, of this function ($E(\boldsymbol{s}^*) = 0$). A dynamical system defined via e.g., a simple gradient descent to find the global minimum of $E(\boldsymbol{s})$, however, will typically get trapped in local minima where $E(\boldsymbol{s}) > 0$. The continuous trajectories approach these attracting non-solution fixed-points at an exponential rate (the vector field is analytic) and hence, a corresponding exponential extraction is needed from these regions by an algorithm that does not get stuck. To achieve that, we modified the energy function by introducing auxiliary variables for each constraint ($a_m \in [1, \infty), m = 1, \ldots, M$), acting similar to Lagrange multipliers: $V(\boldsymbol{s},\boldsymbol{a})=\sum_{m=1}^{M}a_mK_m^2(\mathbf{s})$. The dynamics of $\mathbf{s}$ is defined as a gradient descent on the energy surface and the role of the auxiliary variables is to provide extra dimensions along which the trajectory escapes from local wells. The dynamics ensures that whenever a constraint is not satisfied, the respective auxiliary variable grows *exponentially*, modifying the energy function and ultimately extracting the trajectory from the local minima/wells [28]. Due to the unbounded auxiliary variables exhibiting exponential growth when needed, this system achieves polynomial continuous-time efficiency, however at the cost of exponentially large fluctuations in the energy function $V(\boldsymbol{s},\boldsymbol{a})$. This study has also shown that the hardness of (solvable) problems appears as chaotic dynamics, however, it is of transient type [41,42] as the system still finds the solution.

While using unbounded auxiliary variables one can avoid local traps and achieves polynomial efficiency in the analog search times, the question is whether one can design a continuous-time dynamical system for *k*-SAT using only bounded variables (implementation friendly), but preserving as many of the desirable features of the system as possible. At a recent conference [30] we presented an implementation friendly model for solving *k*-SAT (see below), which is a cellular neural network model similar to those used in CNN computers and also similar to Hopfield models. Here we rigorously define the parameter region where a one-to-one correspondence between solutions and fixed-points can be achieved. We also show that the optimal region of parameters (where the system is most efficient) is independent of the properties of the problem. Most importantly, our system does not get trapped in local minima, finding the solution is one single continuous-time process which does not need "intervention" and tuning of parameters.

Continuous-time asymmetric neural network for k-SAT

Continuous-time recurrent neural networks in general are defined as:

$$\frac{dx_i(t)}{dt}=-x_i(t)+\sum_j w_{ij}f(x_j(t))+u_i \tag{2}$$

where x_i is the state value, or activation potential of the cell, $f(x)$ is the output function of the neuron (usually a sigmoid), u_i is the input, or bias of the neuron and w_{ij} are connection weights. Cellular neural networks have the same form, however in real implementations the cells are placed on a square lattice, and so far only neighbors can influence each other.

We defined our continuous-time asymmetric neural network model on a bipartite graph with two types of nodes (cells) (Fig. 1A) [30]. One type ("*s*-type") represents the variables of *k*-SAT, whose state value will be denoted by s_i, $i=1,\ldots,N$ and their output function defined via (see Fig. 1B):

$$f(s_i)=\frac{1}{2}(|s_i+1|-|s_i-1|). \tag{3}$$

When the Boolean variable is true ($x_i=1$) we will have $f(s_i)=1$, when it is false ($x_i=0$) then $f(s_i)=-1$. However, during the dynamics we allow any continuous value $f(s_i)\in[-1,1]$. For simplicity we say that $f(\boldsymbol{s})$ is a solution of *k*-SAT, whenever $\boldsymbol{x}=[f(\boldsymbol{s})+1]/2$ is a solution. The input of these cells will not be needed, we fix them as $u_i=0\ \forall i$. The self-coupling parameter will be a constant value $w_{ii}=A$, this being one of the important parameters of the model.

The clauses are represented by the second type of cells with state value a_m, $m=1,\ldots,M$ and output function (Fig. 1C):

$$g(a_m)=\frac{1}{2}(1+|a_m|-|1-a_m|). \tag{4}$$

These variables, or "*a*-type" cells will play a similar role as the auxiliary variables in [28,29]. They determine the impact a clause has at a given moment on the dynamics of the $\boldsymbol{s}$ variables. For this reason $g(a_m)=0$ will correspond to the clause being true, and $g(a_m)=1$ to the clause being false. The second important parameter of the model is the self-coupling of these cells $w_{mm}=B$. Their input will be $u_m=u=1-k$ where k represents the number of variables in the clause ($k=3$ for 3-SAT, $k=4$ for 4-SAT, etc.). As we will see later, this is needed in order to achieve the correspondence between *k*-SAT solutions and stable fixed points. The connection weights between the cells are determined by the c_{mi} matrix elements of the given *k*-SAT problem. The dynamical system is defined via:

$$\dot{s}_i(t)=\frac{ds_i(t)}{dt}=-s_i(t)+Af(s_i(t))+\sum_m c_{mi}g(a_m(t)) \tag{5}$$

$$\dot{a}_m(t)=\frac{da_m(t)}{dt}=-a_m(t)+Bg(a_m(t))-\sum_i c_{mi}f(s_i(t))+1-k \tag{6}$$

This neural network is asymmetrical: the influence of a clause on a variable (with connection weight c_{mi}) is exactly the opposite of the influence of the variable on the same clause (with weight $-c_{mi}$). We cannot assign a Lyapunov function (or energy function) to this system, the dynamics is not a simple gradient descent, and in case of hard problems it can show complex chaotic dynamics.

Important theorems

Here we list some important theorems showing the properties of the model. The proofs are presented at the end of the paper (in section Proof of Theorems).

Theorem 1

Variables remain bounded: If initially $|s_i(0)|\le 1$ *and* $0\le a_m(0)\le 1$, *then the state values of cells* $s_i(t)$ *and* $a_m(t)$ *remain bounded for all* $t>0$, $\forall i,m$:

$$|s_i(t)|\le 1+A+\sum_m |c_{mi}| \tag{7}$$

$$-2k\le a_m(t)\le 2+B \tag{8}$$

Theorem 2

Every k-SAT solution has a corresponding stable fixed point: Given a k-SAT formula $\mathcal{F}$, *if* $f(s_i^*)=\pm 1$, $i=1,\ldots,N$ *is a solution of* $\mathcal{F}$ *and* $A>1$, $B>1$ *then the* $(\boldsymbol{s}^*,\boldsymbol{a}^*)$ *point:*

$$s_i^*=Af(s_i^*)\ ,\ a_m^*=-\sum_j c_{mj}f(s_j^*)+1-k \tag{9}$$

$i=1,\ldots,N$, $m=1,\ldots,M$ *is a stable fixed point of the system (5–6).*

Theorem 3

A stable fixed point always corresponds to a solution. If $1<A<2$, $1<B<2[\frac{k}{2}]+2$ *and* $(\boldsymbol{s}^*,\boldsymbol{a}^*)$ *is a stable fixed point, then* $f(\boldsymbol{s}^*)$ *must be a solution of the k-SAT formula. ([·] denotes the integer part.)*

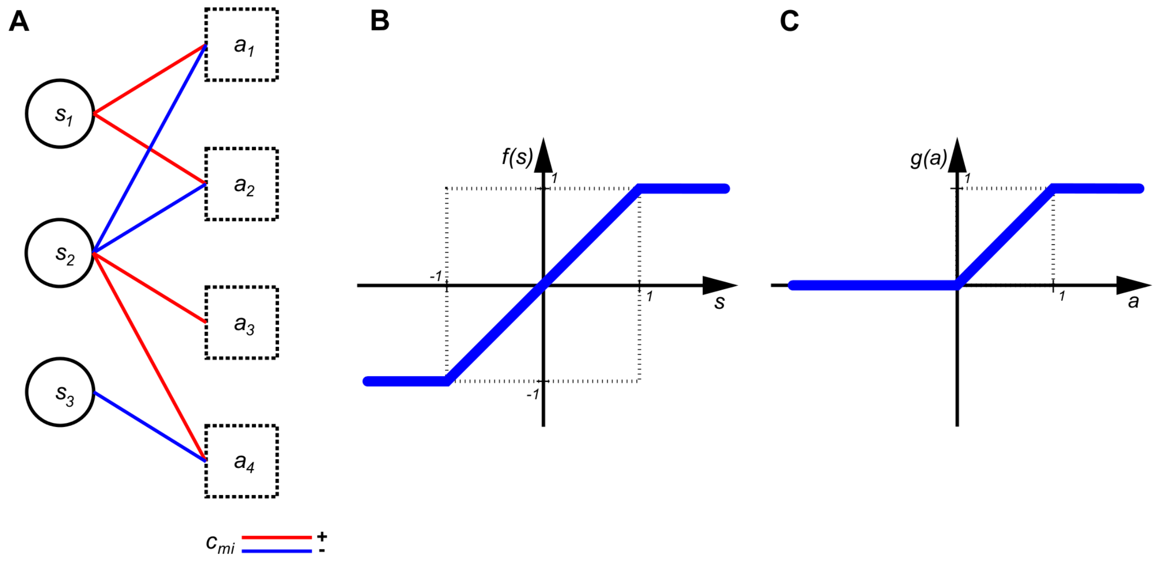

Figure 1. Structure of the neural network. (A) The system is defined on a bipartite graph with two types of nodes. (B) The output function of *s*-type variables $f(s_i) = (|s_i+1| - |s_i-1|)/2$. (C) Output of *a*-type variables $g(a_m) = (1+|a_m| - |1-a_m|)/2$.
doi:10.1371/journal.pone.0073400.g001

When proving this third theorem it was relatively easy to see that it holds for $1 < B < 2$ [30]. Simulation results, however, indicate that the optimal value of parameter B is larger than 2. Indeed, here we present a more general proof showing that the correspondence between stable fixed points and solutions is preserved in the larger interval $2 \leq B < 2\left[\frac{k}{2}\right] + 2$ (see Proof of Theorems).

Numerical results

The theorems presented above guarantee that all stable fixed points of the system correspond to *k*-SAT solutions. However, there is no guarantee that there are no other attractors - such as limit cycles or chaotic attractors - in the system. The existence or non-existence of such attractors is very difficult to show analytically, but simulation results indicate that parameters (A,B) have an optimal region fairly independent of the properties of the problems, where the dynamics avoids getting trapped in non-solution attractors and finds a *k*-SAT solution.

We performed the simulations using the fifth-order adaptive Runge-Kutta method. In Figure 2 we plot the time evolution of a few *s*-type and *a*-type variables and the energy function $E(f(\boldsymbol{s})) = \sum_{m=1}^{M} K_m^2(f(\boldsymbol{s}))$ mentioned above (for details see [28]) for a large 3-SAT problem with $N = 1000$ variables and $\alpha = 4$ constraint density. While our neural network does not explicitly use an energy function (like Hopfield networks do), we use this function to monitor the evolution of the trajectory in its search for a solution. This strongly depends on the parameters (A,B). In Figs. 2A, B, C we show a case when there is transient chaotic dynamics, but finally a solution is found $(A = 1.54, B = 2.18)$. In spite of the fluctuations the energy consistently decreases until finding the solution where $E = 0$. There are parameter values A and B, however, where the dynamics gets trapped in limit cycles, with an example shown in Figs. 2D, E, F $(A = 1.1, B = 1.1)$. In such cases some of the s_i and a_m variables remain constant but others follow complicated periodic orbits. The energy is very noisy in the simulation, but we can see it gets trapped and fluctuates in a narrow interval. This usually happens when the A and B values are small (see also below). In these situations the dynamics gets out very slowly from the subspace $(\boldsymbol{s},\boldsymbol{a}) \in [-1,1]^N \bigcup [0,1]^M$, inside which the dynamics is linear and limit cycles can easily occur. These are not necessarily stable limit cycles, it can happen that the dynamics escapes after a very long time. Similar phenomenon of extremely long transient oscillations have been observed in CNN systems with a particular ring shape [43].

We needed to investigate how the efficiency of the system in finding solutions depends on the (A,B) parameters. As mentioned above the standard way of testing algorithmic performance is to use random SAT instances. In Figs. 3, 4 we show maps covering the $A \in (1,2)$, $B \in (1,3)$ parameter region, depicting the performance of the system and how it changes as the system size varies. We ran 100 random 3-SAT (Fig. 3) and 4-SAT problems (Fig. 4) for each point of the maps (out of the randomly generated instances we use only the satisfiable ones). The resolution of the maps is 0.02. This means that preparing these maps is equivalent to solving $50 \times 100 \times 100 = 5 \times 10^5$ instances for each N. Because these are computationally costly, we had to use relatively small instances. Constraint densities from the hardest regions of 3-SAT $(\alpha = 4.25)$ and 4-SAT $(\alpha = 9.55)$ were used [33–35]. In the first column of Figs. 3, 4 the color indicates the fraction of solved problems in the given time $t_{max} = 5000, \ldots, 15000$. In the second column we show the average continuous-time (not the simulation running time) the system takes to solve them (see color bars). When the solution is not found we include t_{max} in the average. The maps show a peculiar shape consistent while changing the system size. The bottom left corner and the top middle is a parameter region where solutions are hard to find. Our observations also indicate that this is caused by limit cycles in the bottom left corner, and extremely long chaotic transients (super-transients) at the top of the map. In the middle, however there is a large region where the solution is found efficiently. As the system size increases this middle region gets lighter in the first (red) column and darker in the second one (blue). This is because the time needed to solve problems increases with the system size. In order to solve the same fraction of problems (to achieve the same red shade on the map) we would need to greatly increase the simulation times for the larger systems (see also Fig. 7), which is too costly in our case. The statistics being based on only 100 random instances, instead of searching for the

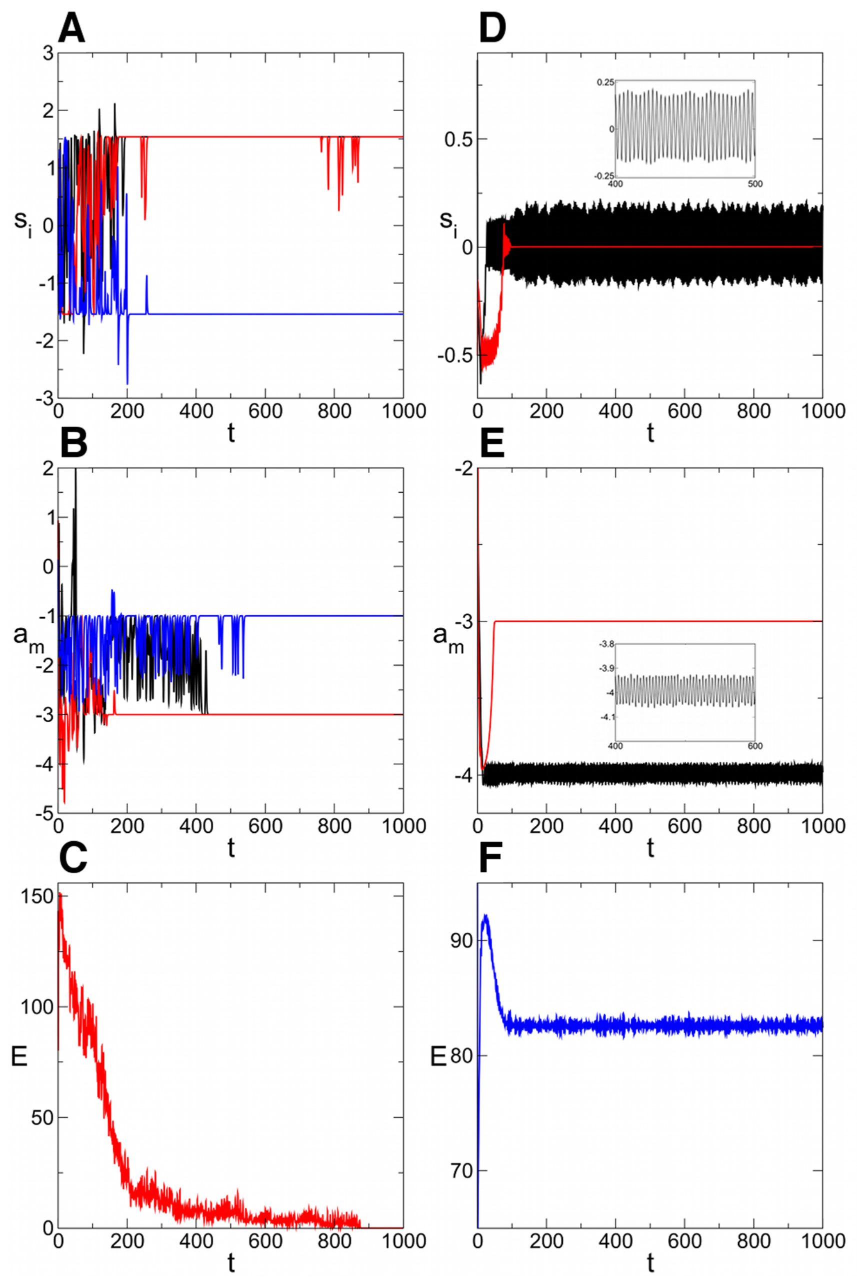

Figure 2. Time evolution of s-type and a-type variables. In a given 3-SAT instance with $N=1000, \alpha=4$ the evolution of three different s_i variables (different colors on (A), (D)), two different a_m variables (B), (E) and the energy of the system (C), (F) is shown for two different parameter settings. (A), (B), (C) A = 1.54, B = 2.18 the solution is found after a chaotic transient. (C), (D), (E) A = 1.1, B = 1.1 the dynamics gets trapped in a limit cycle.
doi:10.1371/journal.pone.0073400.g002

optimal (A,B) parameter setting we indicate the 4% of the whole map (orange squares) where the fraction of solved problems is the largest (when the fractions are identical comparisons are made based on time values). This optimal region is fairly consistent while changing the size of problems (also see Fig. 6A, B).

We also checked how the map and the optimal parameter region changes as varying the constraint density α, and by this the hardness of problems. In the first row of Fig. 5 we show the maps for 3-SAT problems with $N=40$ and $\alpha=3.5,4.0,4.25$ (from left to right). In the last column the frames of the optimal regions of the three maps are placed on each other showing an excellent match. Maps in the second row show results obtained on 4-SAT problems with $N=20$ and $\alpha=8.5,9.0,9.55$ and in the third row on 5-SAT problems with $N=20$ and $\alpha=15,18,20.8$ (the hardest region in 5-SAT being around $\alpha=20.8$ [34]). Again the optimal areas show a good match. Because for larger k the B parameter can have larger values (see Theorem 3), for 5-SAT we show the maps on the $B\in(1,4)$ interval. However, we see that the optimal region still remains at the lower values close to the optimal parameter range of 3-SAT and 4-SAT.

These maps indicate that the optimal parameter region is surprisingly consistent while changing the size (N) and hardness (α) of problems. On Fig. 6A, B, C we draw on top of each other the frames of these optimal areas found on maps of all simulations

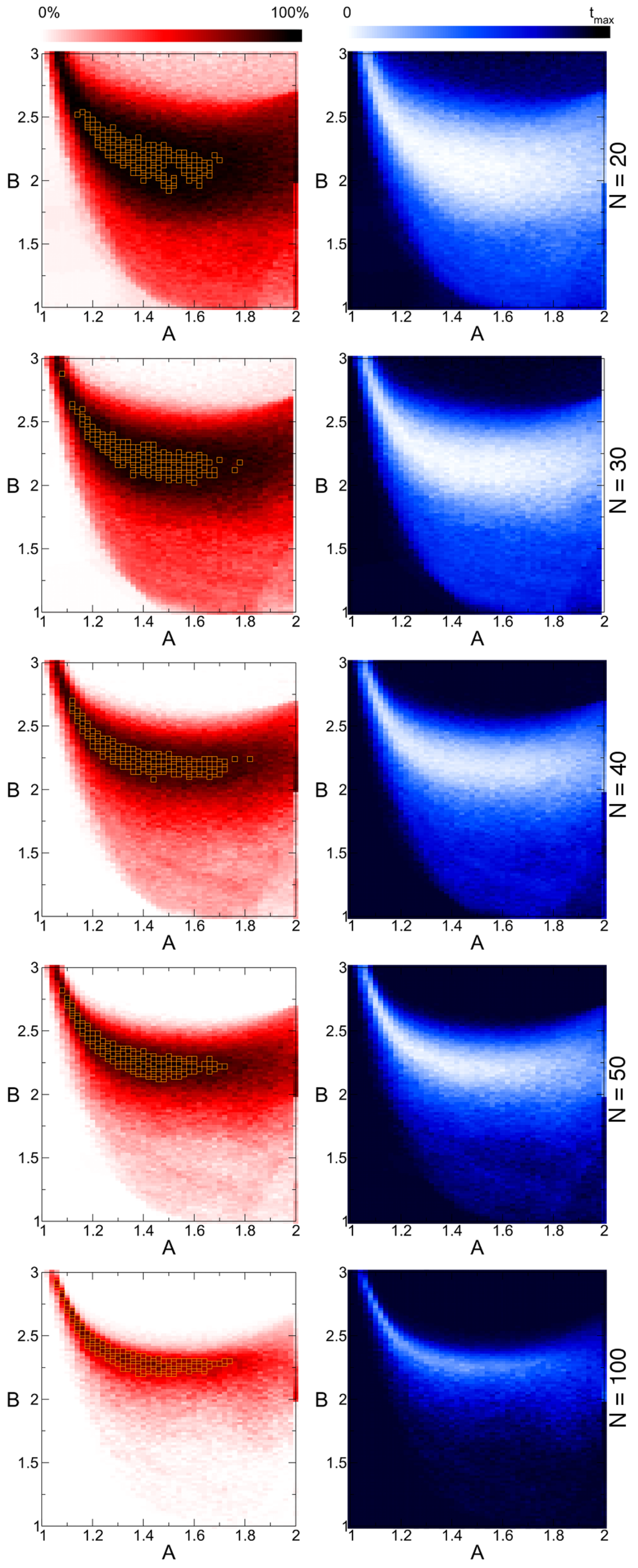
0%
100%
0
t_{max}
B
A
N = 20
N = 30
N = 40
N = 50
N = 100

Figure 3. Parameter dependence of dynamics in 3-SAT problems with critical constraint density. For each point $A \in (1,2), B \in (1,3)$ on the map we solve 100 randomly chosen satisfiable 3-SAT instances, with $N = 20,30,40,50,100$ and $\alpha = 4.25$. Maps in the first column show the fraction of solved problems, in the second column the average continuos-time needed (see color bars). The maximal time $t_{max} = 5 \times 10^3$ for $N = 20$, 10×10^3 for $N = 30$, 15×10^3 for $N = 40$, 20×10^3 for $N = 50$, 25×10^3 for $N = 100$. Orange squares on the red maps indicate 4% of the map with highest efficiency.
doi:10.1371/journal.pone.0073400.g003

performed on 3-SAT, 4-SAT and 5-SAT problems, respectively. On Fig. 6D we compare 3-SAT (black frame), 4-SAT (red) and 5-SAT (green) by showing the optimal regions in case of $N = 20$ and constraint densities $\alpha = 4.25, 9.55, 20.8$ these being the hardest SAT phases for each. This indicates that there is a smaller region (in the middle) which seems to be part of the optimal regions of all k.

Because the variables remain bounded and there is no extra energy introduced into the system (contrary to the system in [28]) the dynamics naturally has an exponential continuous-time complexity in the hard-SAT region. In Fig. 7A, B, C we plot the fraction $p(t)$ of problems which remain unsolved after a time t for various sizes ($N = 20,30,\ldots,100$) of randomly chosen 3-SAT, 4-SAT and 5-SAT instances with constraint densities in the hardest α regions. We chose a parameter setting ($A = 1.4, B = 2.24$) from the common part of the optimal regions shown in Fig. 6D. The distributions are decreasing as a power law ($p(t) \sim t^{-\beta(k,N)}$), where the power $\beta(k,N)$ depends on k and the size N of the k-SAT instances. $\beta(N)$ is again a power law, $\beta(N) \sim N^{-\gamma}$ ($\gamma \cong 0.95, 1.76, 2.23$ for $k = 3,4,5$) indicating that the time complexity of the model is exponential for solving a fixed fraction of problems (Fig. 7D). This power-law decrease of $p(t)$ shows that the probability of not finding the solution goes to zero (not a positive constant), supporting the claim that in this optimal parameter region (common for all k) the dynamics does not get trapped in limit cycles and a solution is always found after a transiently chaotic period.

Discussion

Solving NP-complete problems is a key test for any non-conventional computation. Here we presented an asymmetric continuous-time neural network that can efficiently solve Boolean satisfiability without getting trapped in non-solution attractors, and without requiring careful parameter tuning during the dynamical process. In particular, it has the following key properties: 1) It has a deterministic continuous-time dynamics. 2) All variables remain bounded. 3) The dynamics can be implemented with analog circuits (has almost the same form as used in CNN computers). 4) There is a one-to-one correspondence between the stable fixed-points of the system and the solutions of the k-SAT problem.

Numerical simulations show that our method works consistently and efficiently on 3-SAT, 4-SAT and 5-SAT problems. A careful study of the dynamics as function of the two important parameters of the system shows that their optimal interval has a peculiar shape surprisingly consistent when changing the size and hardness of SAT instances. Comparing the optimal parameter regions for the different $k = 3,4,5$ SAT classes we find a common parameter range which seems to work efficiently for each k-SAT instance. This assures that the system does not need careful choosing of parameters depending on the properties of SAT formulae.

While there are parameter intervals where limit cycles frequently occur (mainly the smaller values of A and B), statistics done with a parameter setting (the same for all k) chosen from the optimal region shows that here the dynamics does not get trapped in long cycles. The distribution of transient times shows a clear power-law decay in contrast with the distributions obtained with non-optimal parameters (not shown on the figures), where it goes to a positive constant value indicating that a part of problems are not solved because the dynamics gets trapped in long oscillations.

Previous approaches mainly concentrated on symmetric - and dominantly on discrete-time - neural networks. Most of the time gradient descent dynamics was used possibly combined with annealing processes. Here we have shown that *asymmetric continuous-time* neural networks can be designed to solve constraint satisfaction problems on their own, without additional annealing processes. This dynamics does not get trapped in non-solution attractors and transient chaotic behavior appears as an unavoidable byproduct of optimization hardness [28,29].

On an analog device this algorithm would take a single operation: the connection weights are based on the c_{mi} matrix corresponding to the given k-SAT instance (the input of the operation) and starting from any initial condition the system searches until finding a solution, without the need of any further intervention by the user.

Our model is implementation friendly, being similar to neural networks used in analog CNN computers. However, when considering analog computation, an important question - which needs to be investigated - is the effect of noise on the dynamics. Preliminary studies show that similarly to other transiently chaotic systems [41,42], the $p(t)$ distribution of transient times (and thus the efficiency of finding the solution) is not sensitive to noise. Actually noise effects may even help avoiding long transient oscillations, thus extending the optimal parameter region.

Proof of Theorems

Proof of Theorem 1

Let us recall the dynamics of $s_i(t)$:

$$\frac{ds_i(t)}{dt} = -s_i(t) + Af(s_i(t)) + \sum_m c_{mi} g(a_m(t)) \tag{10}$$

This is a first-order ODE and its solution can be written as:

$$s_i(t) = s_i(0)e^{-t} + \int_0^t e^{-(t-\tau)} \left(Af(s_i(\tau)) + \sum_m c_{mi} g(a_m(\tau)) \right) d\tau \tag{11}$$

It follows that

$$|s_i(t)| \le |s_i(0)e^{-t}| + \left| \int_0^t e^{-(t-\tau)} \left(Af(s_i(\tau)) + \sum_m c_{mi} g(a_m(\tau)) \right) d\tau \right| \tag{12}$$

$$\le |s_i(0)|e^{-t} + \int_0^t e^{-(t-\tau)} \left| Af(s_i(\tau)) + \sum_m c_{mi} g(a_m(\tau)) \right| d\tau \tag{13}$$

$$\le |s_i(0)|e^{-t} + \int_0^t e^{-(t-\tau)} \left(A|f(s_i(\tau))| + \sum_m |c_{mi} g(a_m(\tau))| \right) d\tau \tag{14}$$

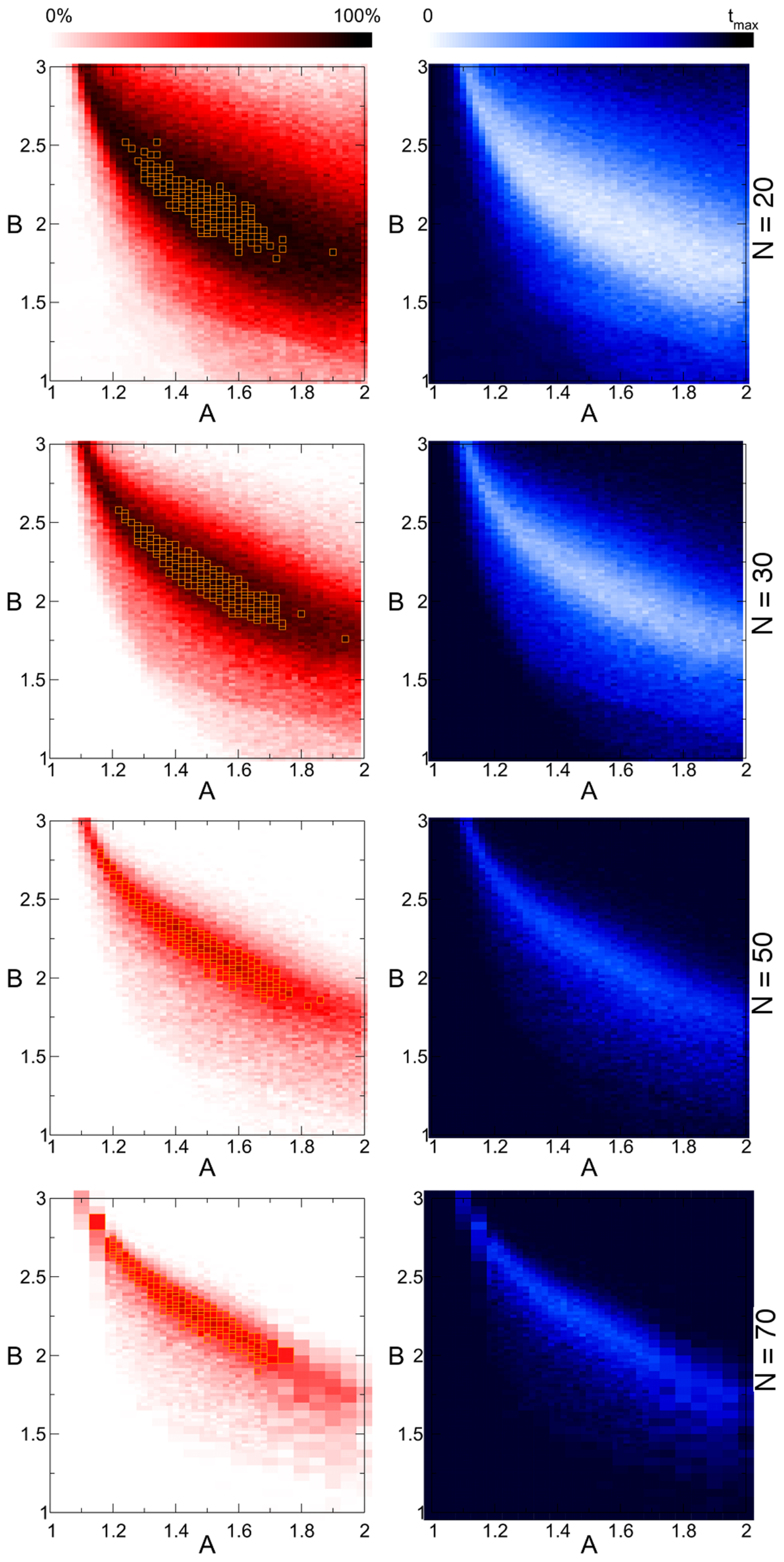
0%
100%
0
t_{max}
B
A
N = 20
N = 30
N = 50
N = 70

Figure 4. Parameter dependence of dynamics in 4-SAT problems with critical constraint density. For each point $A\in(1,2), B\in(1,3)$ on the map we solve 100 randomly chosen satisfiable 4-SAT instances, with $N=20,30,50,70$ and $\alpha=9.55$. Maps in the first column show the fraction of solved problems, in the second column the average continuos-time needed (see color bars). The maximal time $t_{max}=5\times10^3$ for $N=20,30,50$ and $t_{max}=15\times10^3$ for $N=70$. Orange squares on the red maps indicate 4% area of the map with highest efficiency.
doi:10.1371/journal.pone.0073400.g004

$$\le |s_i(0)|e^{-t}+\left(A+\sum_m |c_{mi}|\right)\int_0^t e^{-(t-\tau)}d\tau \qquad (15)$$

$$\le |s_i(0)|+A+\sum_m |c_{mi}| \qquad (16)$$

$$\le 1+A+\sum_m |c_{mi}| \qquad (17)$$

where we used the facts that $|f(s_i)|\le 1$, $0\le g(a_m)\le 1$ and the initial condition $|s_i(0)|\le 1$. It is also easy to see that $0\le\int_0^t e^{-(t-\tau)}d\tau\le 1$.

For proving Eq. (8) we recall the dynamical equation:

$$\frac{da_m(t)}{dt}=-a_m(t)+Bg(a_m(t))-\sum_i c_{mi}f(s_i(t))+1-k \qquad (18)$$

which has the solution:

$$a_m(t)=a_m(0)e^{-t}+\int_0^t e^{-(t-\tau)}\left(Bg(a_m(\tau))-\sum_i c_{mi}f(s_i(\tau))+1-k\right)d\tau \qquad (19)$$

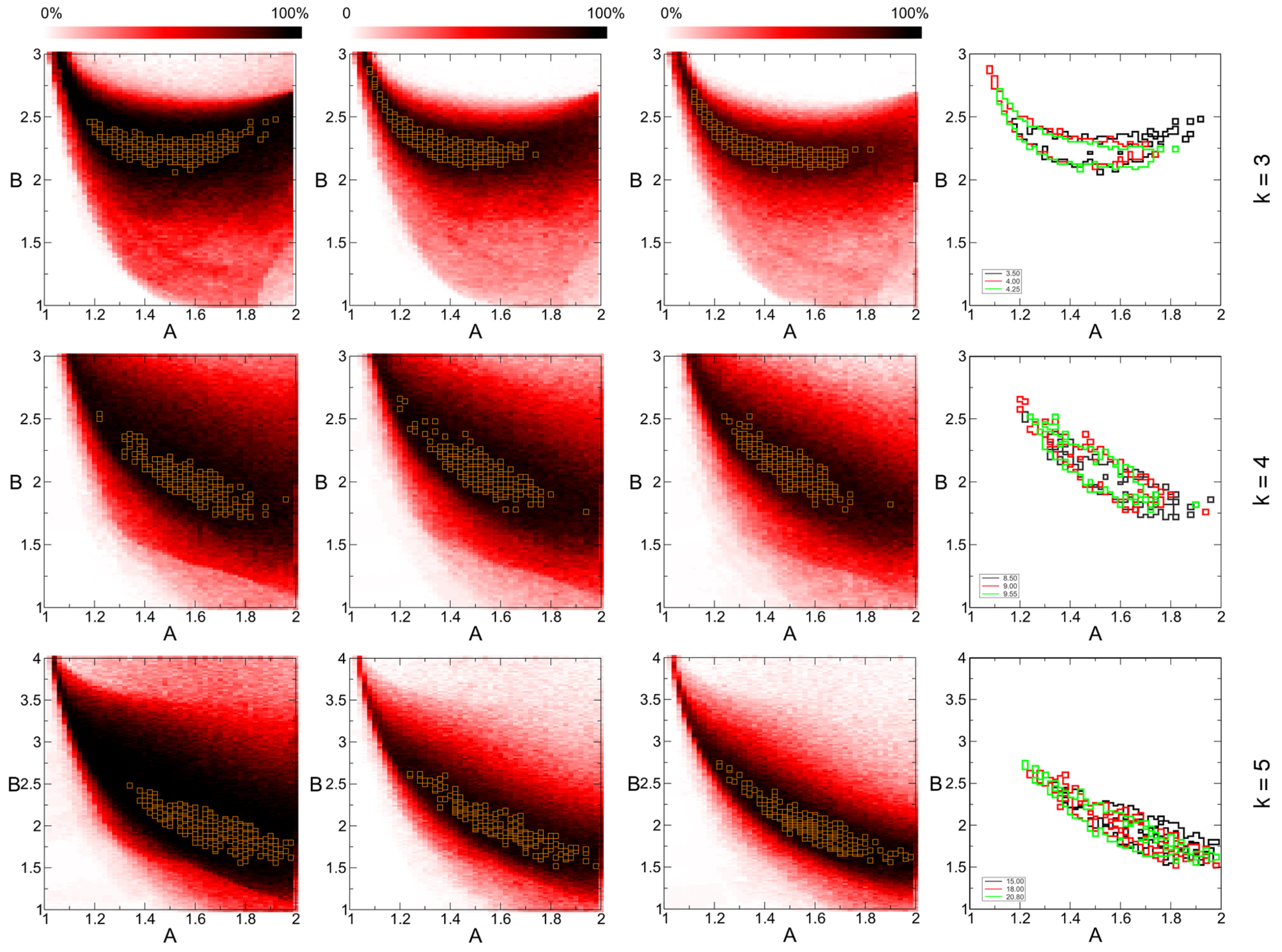

Figure 5. Parameter dependence of dynamics in 3-SAT, 4-SAT and 5-SAT problems with fixed size and varying constraint density. For each (A,B) on the map we solve 100 randomly chosen satisfiable instances. The color indicates the fraction of solved problems (see color bar). Simulations were performed on 3-SAT problems (first row) with $N=40$ and constraint densities $\alpha=3.5,4.0,4.25$ (left to right), 4-SAT (second row) with $N=20$ and $\alpha=8.5,9.0,9.55$, and 5-SAT (third row) with $N=20$ and $\alpha=15,18,20.80$. The optimal parameter regions are shown with orange squares on the color maps. In the last column we compare the optimal regions of the three maps in each particular row (black, red, green from left to right), by drawing the frames of these regions.
doi:10.1371/journal.pone.0073400.g005

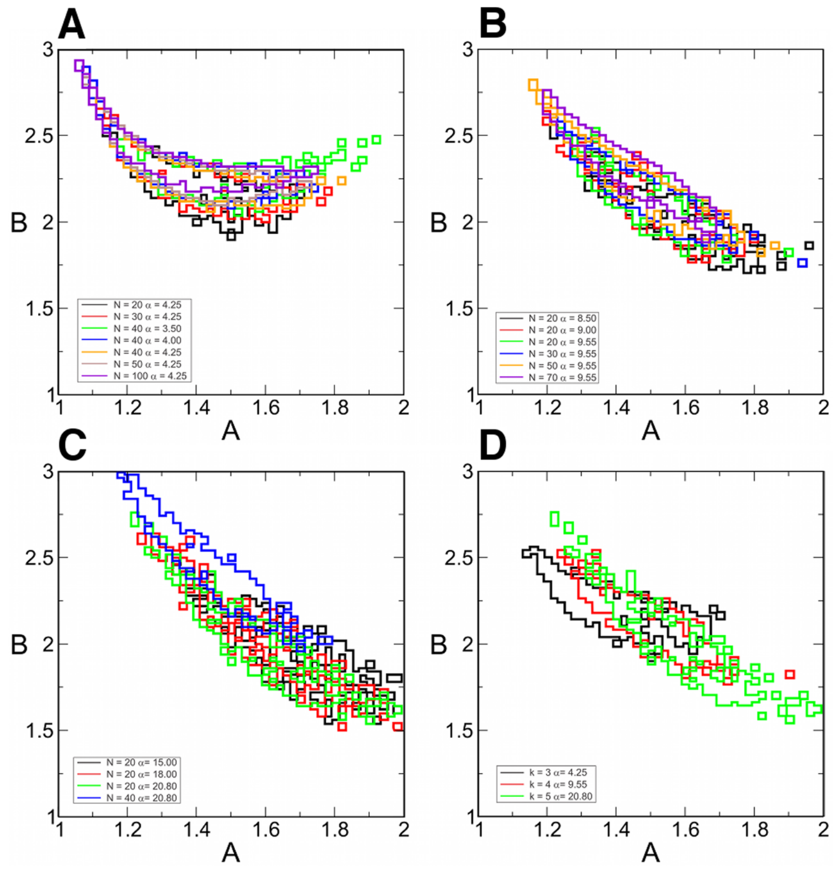

Figure 6. Consistency of the optimal parameter region. The optimal (*A*,*B*) parameter regions are shown with different colors for all maps obtained for A) 3-SAT, B) 4-SAT, C) 5-SAT instances. D) We compare the maps for different $k=3,4,5$ by choosing $N=20$ and the hard-SAT phase for each *k*: $\alpha=4.25$ in 3-SAT (black), $\alpha=9.55$ in 4-SAT (red) and $\alpha=20.8$ in 5-SAT (green).
doi:10.1371/journal.pone.0073400.g006

First we will prove the rhs of Eq. (8).

$$a_m(t) \le a_m(0)e^{-t} + \int_0^t e^{-(t-\tau)} \left| Bg(a_m(\tau)) - \sum_i c_{mi} f(s_i(\tau)) + 1 - k \right| d\tau \tag{20}$$

$$\le a_m(0)e^{-t} + \int_0^t e^{-(t-\tau)} \left(B|g(a_m(\tau))| + \sum_i |c_{mi} f(s_i(\tau))| + 1 - k \right) d\tau \tag{21}$$

Because there are exactly k variables in a clause and $|f(s_i)| \le 1$, then $\sum_i |c_{mi} f(s_i(\tau))| \le k$. Recall also that $g(a_m) \le 1$, and the initial condition $a_m(0) \le 1$. Using this we can write:

$$a_m(t) \le a_m(0) + (B+1) \le 2 + B \tag{22}$$

For proving the lhs of Eq. (8) we will use that $B > 0$, $g(a_m) \ge 0$, $a_m(0) \ge 0$, $\left|\sum_i c_{mi} f(s_i(\tau))\right| \le k$ and $0 \le \int_0^t e^{-(t-\tau)} d\tau \le 1$:

$$\begin{aligned} a_m(t) &\ge a_m(0)e^{-t} + \int_0^t e^{-(t-\tau)} \left(Bg(a_m(\tau)) - \left| \sum_i c_{mi} f(s_i(\tau)) \right| + 1 - k \right) d\tau \\ &\ge a_m(0)e^{-t} + \int_0^t e^{-(t-\tau)} (-k + 1 - k) d\tau \\ &\ge -2k. \end{aligned} \tag{23}$$

Proof of Theorem 2

We presented this proof in the conference paper [30], however we briefly recall it here to make it easier to readers to follow the next proof.

Given the definitions, it follows that if $f(s_i^*)$ satisfies the clause C_m then $c_{mi} f(s_i^*) = 1$ and if it does not satisfy it, when $c_{mi} \ne 0$, then $c_{mi} f(s_i^*) = -1$. Accordingly, the sum $\sum_i c_{mi} f(s_i^*)$ in Eq. (9) can take $k+1$ possible values: $-k, -k+2, \ldots, k-2, k$. Only the value of $-k$ corresponds to the clause C_m not being satisfied, in all other cases there is at least one variable satisfying the constraint. Because by assumption $f(s_i^*)$ is a solution, we must have:

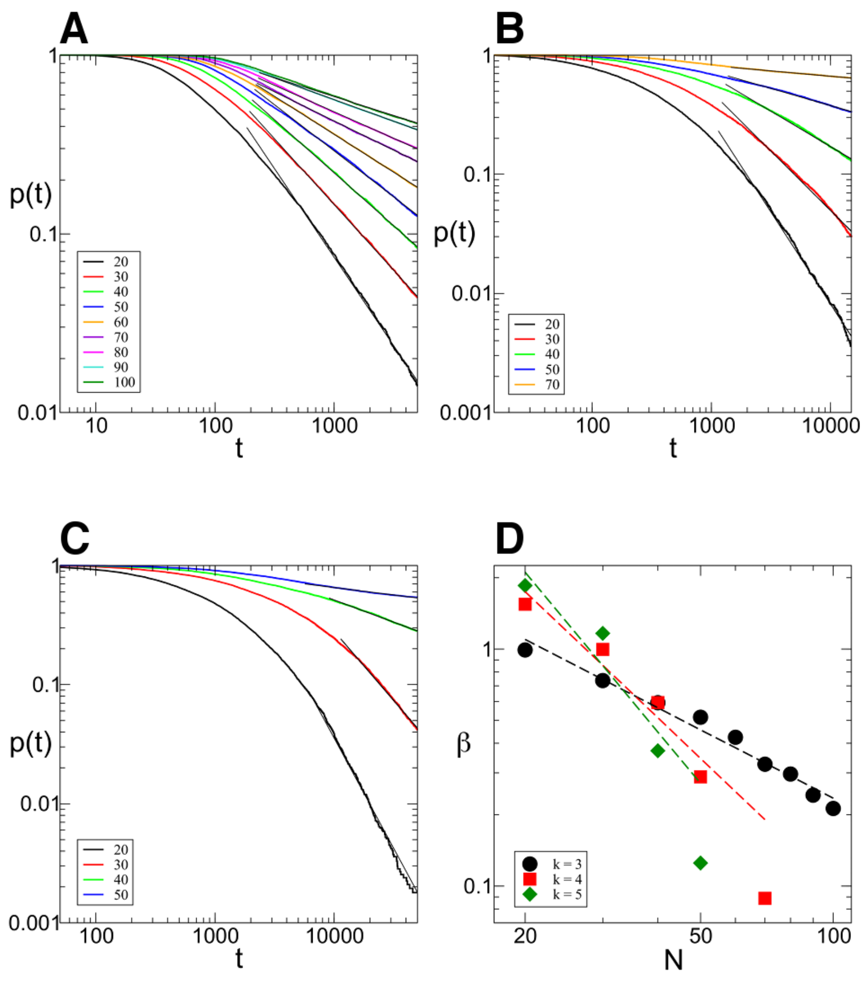

Figure 7. Distribution of transient times. The number of k-SAT problems $p(t)$ which remain unsolved as function of the continuous-time t of the system, for A) $k=3$, $\alpha=4.25$, B) $k=4$, $\alpha=9.55$ C) $k=5$, $\alpha=20.8$ for different values of N (see the legends). The statistics was made on 10^4 problems for each k and N. The last part of the distributions are fitted with a power law $p(t)\sim t^{-\beta(k,N)}$, and D) shows the dependence of the exponent β on N. For $k=3$ this can be fitted with a power law $\beta(N)\sim N^{-0.95}$. For 4-SAT and 5-SAT the β values for larger N are not precise (statistics would be needed on much longer time interval), but the exponents are expected to be around -1.76 and -2.23.
doi:10.1371/journal.pone.0073400.g007

$$\begin{aligned}&\sum_j c_{mj} f(s_j^*) \ge -k+2 \Rightarrow \\ &\Rightarrow a_m^* = -\sum_m c_{mi} f(s_i^*) + 1 - k \le -1 \overset{(4)}{\Rightarrow} \\ &\Rightarrow g(a_m^*) = 0,\ \forall\, m = 1,\dots,M\end{aligned} \qquad (24)$$

Using Eqs. (9, 24) and including the values of s_i^*, a_m^*, $g(a_m^*)$ into the dynamical equations (5) and (6) we get $ds_i^*/dt=0$ and $da_m^*/dt=0$, confirming that we have a fixed point.

To prove stability, we will show the following: Starting in any point $(\boldsymbol{s},\boldsymbol{a}) = (\boldsymbol{s}^* + \boldsymbol{\epsilon}, \boldsymbol{a}^* + \boldsymbol{\delta})$ in a compact vicinity of the fixed point, such that $|s_i - s_i^*| = |\epsilon_i| < \epsilon < A-1\ \forall i$ and $|a_m - a_m^*| = |\delta_m| < \delta < 1\ \forall m$, the square distance from the fixed point:

$$R(\boldsymbol{s(t)},\boldsymbol{a(t)}) = \sum_i (s_i(t) - s_i^*)^2 + \sum_m (a_m(t) - a_m^*)^2 \qquad (25)$$

is decreasing, that is $dR(\boldsymbol{s(t)},\boldsymbol{a(t)})/dt < 0$, until the dynamics reaches the fixed point where $dR(\boldsymbol{s(t)},\boldsymbol{a(t)})/dt = 0$.

From Eq. (9) we know that $|s_i^*| = A|f(s_i^*)| = A > 1\,\forall i$. We have two cases: 1) If $s_i^* = A$ and $f(s_i^*) = 1$ then $s_i^* + \epsilon_i > A - (A-1) = 1 \Rightarrow f(s_i^* + \epsilon_i) = f(s_i^*) = 1$. 2) If $s_i^* = -A$ and $f(s_i^*) = -1$ then $s_i^* + \epsilon_i < -A + (A-1) = -1 \Rightarrow f(s_i^* + \epsilon_i) = f(s_i^*) = -1$. In both cases $f(s_i^* + \epsilon_i) = f(s_i^*)$.

Similarly from Eq. (24) $a_m^* \le -1$ and condition $\delta < 1$ it follows that $a_m^* + \delta_m < 0 \Rightarrow g(a_m^* + \delta_m) = g(a_m^*) = 0$.

Inserting these in Eqs. (5, 6) and using Eq. (9) it can be easily seen that $\dot{s}_i = -s_i + Af(s_i^*) = -s_i + s_i^* = -\epsilon_i$ and $\dot{a}_m = -a_m - \sum_i c_{mi} f(s_i^*) + 1 - k = -a_m + a_m^* = -\delta_m$ and the derivative of the distance is:

$$\begin{aligned}\frac{dR(\boldsymbol{s},\boldsymbol{a})}{dt} &= 2\sum_i \epsilon_i \dot{s}_i + 2\sum_m \delta_m \dot{a}_m = \\ &= -2\sum_i \epsilon_i^2 - 2\sum_m \delta_m^2 \le 0.\end{aligned} \qquad (26)$$

Because the distance from the fixed point cannot increase along any of the axes: $d\epsilon_i^2/dt = -\epsilon_i^2$, $d\delta_i^2/dt = -\delta_i^2$, the conditions set for ϵ_i, δ_i remain valid for all i,m and the distance continues to decrease until the dynamics reaches the fixed point where (and only there) $dR(\boldsymbol{s},\boldsymbol{a})/dt=0$.

Proof of Theorem 3

It is easy to see that similarly to CNN models our system has the following property (the proof for CNN can be found in [12]): if $A>1$ then in a stable fixed point $|s_i^*|>1$ ($|f(s_i^*)|=1$) $\forall i$; if $B>1$ then $a_m^*<0$ or $a_m^*>1$ ($g(a_m^*)\in\{0,1\}$) $\forall m$. (If these conditions do not hold, there is always an unstable direction along which the dynamics can escape from the fixed point, see [12].)

Being in a stable fixed point $\mathbf{s}^*$:

$$\dot{s}_i^*=-s_i^*+Af(s_i^*)+\sum_m c_{mi}g(a_m^*)=0,\ \forall i=1,\ldots,N \tag{27}$$

Multiplying Eq. (27) with $f(s_i^*)$ we get:

$$\begin{aligned}&-s_i^*f(s_i^*)+Af(s_i^*)^2+\sum_m c_{mi}f(s_i^*)g(a_m^*)=0\Leftrightarrow\\&-|s_i^*|+A+\sum_{c_{mi}f(s_i^*)=1}g(a_m^*)-\sum_{c_{ni}f(s_i^*)=-1}g(a_n^*)=0\Leftrightarrow\\&|s_i^*|=A+J_i^{(+)}-J_i^{(-)}>1\Leftrightarrow\\&J_i^{(-)}-J_i^{(+)}<A-1\Leftrightarrow\\&J_i^{(-)}-J_i^{(+)}<1\end{aligned} \tag{28}$$

where we used $A\le 2$ and introduced the notation $J_i^{(+)}$ and $J_i^{(-)}$ for the two parts of the sum, the first (second) part includes the clauses which are satisfied (not satisfied) by variable s_i^*.

As discussed in the previous theorem, for an unsatisfied clause C_m the sum $\sum_i c_{mi}f(s_i^*)=-k$. Inserting this into the dynamical equation (6) if in the fixed point we have an unsatisfied clause C_m, then:

$$\begin{aligned}\dot{a}_m^*&=-a_m^*+Bg(a_m^*)-\sum_i c_{mi}f(s_i^*)+1-k\\&=-a_m^*+Bg(a_m^*)+1=0.\end{aligned} \tag{29}$$

Because $g(a_m^*)$ must be 0 or 1 and $B>1$, this can hold if and only if $g(a_m^*)=1$ and $a_m^*=B+1$ for unsatisfied constraints. When parameter $B<2$ it can be shown that contrary to unsatisfied clauses, satisfied clauses *must* have $g(a_m^*)=0$ (see [30]). However, this second statement is no longer true when B can have values larger than 2: there *can be* satisfied constraints for which $g(a_m^*)=1$. So let us denote as $M^{(q)}$ ($q\in\mathbf{Z},0\le q\le k$), the number of clauses for which $g(a_m^*)=1$ and there are exactly q variables satisfying the clause. (Here $M^{(0)}$ is exactly the number of unsatisfied constraints.) If there are $M^{(q)}$ clauses with $g(a_m^*)=1$ and satisfied by exactly q variables, from the definitions introduced in Eq. (28) we get:

$$\sum_i J_i^{(+)}=\sum_i\sum_{c_{mi}f(s_i^*)=1}g(a_m^*)=\sum_{q=1}^k qM^{(q)} \tag{30}$$

$$\sum_i J_i^{(-)}=\sum_i\sum_{c_{mi}f(s_i^*)=-1}g(a_m^*)=\sum_{q=0}^k (k-q)M^{(q)} \tag{31}$$

It follows that:

$$\begin{aligned}&\sum_i J_i^{(-)}-\sum_i J_i^{(+)}=kM^{(0)}+\sum_{q=1}^k(k-2q)M^{(q)}=\\&=kM^{(0)}+\sum_{1\le q\le k/2}(k-2q)M^{(q)}-\sum_{k/2<q\le k}(2q-k)M^{(q)}\end{aligned} \tag{32}$$

We will show that the second sum is zero. If we have a clause satisfied by $q>k/2$ variables, q being an integer this is equivalent with the condition $q\ge\left[\frac{k}{2}\right]+1$, where [.] denotes the integer part of the number. Using the boundaries defined for parameter B it follows that $q\ge\left[\frac{k}{2}\right]+1>\frac{B}{2}$. Using again that we are in a fixed point (Eq. (29))

$$\dot{a}_m^*=-a_m^*+Bg(a_m^*)-\sum_i c_{mi}f(s_i^*)+1-k=0\Rightarrow \tag{33}$$

$$\begin{aligned}a_m^*&=Bg(a_m^*)-\sum_i c_{mi}f(s_i^*)+1-k=\\&=Bg(a_m^*)-(-k+2q)+1-k=\\&=Bg(a_m^*)-2q+1<2qg(a_m^*)-2q+1\Leftrightarrow a_m^*<1\end{aligned} \tag{34}$$

where we used again that the clause is satisfied by exactly q variables, so $\sum_i c_{mi}f(s_i^*)=q-(k-q)=-k+2q$. Because of this inequality (34) we cannot have $g(a_m^*)=1$ and it follows that $M^{(q)}=0$. In Eq.(32) the negative sum disappears and we have:

$$\sum_i J_i^{(-)}-\sum_i J_i^{(+)}=kM^{(0)}+\sum_{1\le q\le k/2}(k-2q)M^{(q)}\ge kM^{(0)} \tag{35}$$

Because $M^{(0)}$ equals the number of unsatisfied clauses, if $f(\boldsymbol{s}^*)$ is not a solution, then $M^{(0)}\ge 1\Rightarrow$

$$\sum_i J_i^{(-)}-\sum_i J_i^{(+)}\ge k \tag{36}$$

Because the values of $J_i^{(-)}$ and $J_i^{(+)}$ are non-negative integers, and $k>1$ it follows that there must be at least one value j such that:

$$J_j^{(-)}-J_j^{(+)}\ge 1 \tag{37}$$

contradicting condition (28). This means that if $f(\boldsymbol{s}^*)$ is not a solution of $\mathcal{F}$ it cannot be a fixed point.

Acknowledgments

We thank Zoltán Toroczkai and Zoltán Néda for the useful discussions and a critical reading of the manuscript.

Author Contributions

Analyzed the data: BM. Wrote the paper: MER. Performed the computer simulations: BM. Designed the mathematical model: MER. Proved the theorems: MER.

References

1. Solomon P (2000) Device innovation and material challenges at the limits of CMOS technology. An Rev of Materials Science 30: 681–697.
2. The International Technology Roadmap for Semiconductors (2013) Available: http://www.itrs.net.Accessed 2013 Aug 16.
3. Branicky M (1994) Analog computation with continuous ODEs. Workshop on Physics and Computation, Dallas TX USA : 265–274.
4. Siegelmann H, Sontag E (1994) Analog computation via neural networks. Theoretical Computer Science 131: 331–360.
5. Moore C (1996) Recursion theory on the reals and continuous-time computation. Theoretical Computer Science 162: 23–44.
6. Ben-Hur A, Siegelmann H, Fishman S (2002) A theory of complexity for continuous time systems. Journal of Complexity 18: 51–86.
7. Siegelmann H (1995) Computation beyond the Turing limit. Science 268: 545–548.
8. Sima J, Orponen P (2003) Continuous-time symmetric Hopfield nets are computationally universal. Neur Comp. 15: 693–733.
9. Sima J, Orponen P (2003) General-purpose computation with neural networks: A survey of complexity theoretic results. Neur Comp. 15: 2727–2778.
10. Orponen P (1992) Neural Networks and Complexity Theory. Lect Notes in Computer Science 629: 50–61.
11. Orponen P (1997) The computational power of continuous time neural networks. Lect Notes in Computer Science 1338: 86–103.
12. Chua L, Yang L (1988) Cellular neural networks - theory. IEEE Trans on Circuits and Systems I 35: 1257–1272.
13. Roska T, Chua L (1993) The CNN Universal Machine—An Analogic Array Computer. IEEE Trans on Circuits and Systems II 40: 163–173.
14. Liu S-C, Kramer J, Indiveri G, Delbrück T, Douglas R (2002) Analog VLSI - Circuits and Principles. Cambridge, MA: MIT Press.
15. Douglas R, Mahowald M, Mead C (1995) Neuromorphic Analog VLSI. Annual Rev of Nuroscience 18: 255–281.
16. Chua L, Roska T, Venetianer P (1993) The CNN is as universal as the Turing Machine. IEEE Trans on Circuits and Systems I - Fund Theory and App. 40: 289–291.
17. Hopfield J (1984) Neurons with graded response have collective computational properties like those of 2-state neurons. Proc of the Nat Ac of Sci of the United States of America - Bio Scie. 81: 3088–3092.
18. Hopfield J, Tank D (1985) Neural Computation of Decisions in Optimization Problems. Bio Cyber. 52: 141–152.
19. Hopfield J, Tank D (1986) Computing with Neural Circuits—A model. Science 233: 625–633.
20. Ercsey-Ravasz M, Roska T, Néda Z (2009) Cellular Neural Networks for NP-Hard Optimization. Eurasip J on Advances in Signal Proc. doi:10.1155/2009/646975.
21. Ercsey-Ravasz M, Roska T, Néda Z (2009) Stochastic optimization of spin-glasses on cellular neural/nonlinear network based processors. Physica A - Statistical Mechanics and its Applications 388: 1024–1030.
22. Ercsey-Ravasz M, Roska T, Néda Z (2008) Statistical physics on cellular neural network computers. Physica D - Nonlinear Phenomena 237: 1226–1234.
23. Chen L, Aihara K (1995) Chaotic Simulated Annealing by a Neural-Network Model with Transient Chaos. Neur Net. 8: 915–930.
24. Chen L, Aihara K (1997) Chaos and asymptotical stability in discrete-time neural networks. Physica D 104: 286–325.
25. Wang L, Smith K (1998) On chaotic simulated annealing. IEEE Trans. on Neur Net. 9: 716–718.
26. Wang L, Li S, Tian F, Fu X (2004) A noisy chaotic neural network for solving combinatorial optimization problems: Stochastic chaotic simulated annealing. IEEE Trans. on Sys. MAN and Cybernetics Part B - Cybernetics 34: 2119–2125.
27. Kwok T, Smith K (1999) A unified framework for chaotic neural-network approaches to combinatorial optimization. IEEE Trans on Neur Net. 10: 978–981.
28. Ercsey-Ravasz M, Toroczkai Z (2011) Optimization hardness as transient chaos in an analog approach to constraint satisfaction. Nature Physics 7: 966–970.
29. Ercsey-Ravasz M, Toroczkai Z (2012) The Chaos Within Sudoku. Scientific Reports 2: 725.
30. Molnár B, Toroczkai Z, Ercsey-Ravasz M (2012) Continuous-time Neural Networks Without Local Traps for Solving Boolean Satisfiability. 13th Int. Workshop on Cellular Nanoscale Networks and their Applications (CNNA), Turin, IT, Aug 29–31, 2012.
31. Cook SA (1971) The complexity of theorem-proving procedures. ACM Symposium on Theory of Computing (STOC). pp. 151–158. doi: 10.1145/800157.805047.
32. Garey MR, Johnson DS (1979) Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences), 1st edition. Stuttgart, Germany: W. H. Freeman & Co Ltd.
33. Kirkpatrick S, Selman B (1994) Critical-Behavior in the Satisfiability of Random Boolean Expressions. Science 264: 1297–1301.
34. Krzakala F, Montanari A, Ricci-Tersenghi F, Semerijan G, Zdeborova L (2007) Gibbs states and the set of solutions of random constraint satisfaction problems. PNAS 104: 10318–10323.
35. Krzakala F, Zdeborova L (2008) Phase Transitions and Computational Difficulty in Random Constraint Satisfaction Problems. Journal of Physics: Conference Series 95: 012012.
36. Gu J (1994) Global optimization for satisfiability (SAT) problem. IEEE Trans on Knowledge and Data Engineering 6: 361–381.
37. Gu J, Gu QP, Du DZ (1999) On optimizing the satisfiability (SAT) problem. J Comput Sci Technol. 14: 1–17.
38. Nagamatu M, Yanaru T (1996) On the stability of Lagrange programming neural networks for satisfiability problems of propositional calculus. Neurocomputing 13: 119–133.
39. Wah B, Chang Y (1997) Trace-based methods for solving nonlinear global optimization and satisfiability problems. J of Global Optimization 10: 107–141.
40. Wah B, Wang T, Shang Y, Wu Z (2000) Improving the performance of weighted Lagrange-multiplier methods for nonlinear constrained optimization. Information Sciences 124: 241–272.
41. Lai YC, Tél T (2011) Transient Chaos: Complex Dynamics on Finite-Time Scales. Berlin: Springer.
42. Tél T, Lai YC (2008) Chaotic transients in spatially extended systems. Physics Reports 460: 245–275.
43. Forti M, Garay B, Koller M, Pancioni L (2012) An Experimental Study on Long Transient Oscillations in Cooperative CNN Rings. 13th Int. Workshop on Cellular Nanoscale Networks and their Applications (CNNA), Turin, IT, Aug 29–31, 2012.