IEICE Proceeding Series

Amoeba-inspired SAT Solver

Masashi Aono, Song-Ju Kim, Liping Zhu, Makoto Naruse, Motoichi Ohtsu, Hirokazu Hori, Masahiko Hara

Vol. 1 pp. 586-589
Publication Date: 2014/03/17
Online ISSN: 2188-5079

2012 International Symposium on Nonlinear Theory and its Applications
NOLTA2012, Palma, Majorca, Spain, October 22-26, 2012

Amoeba-inspired SAT Solver

Masashi Aono†, Song-Ju Kim†, Liping Zhu‡, Makoto Naruse£,
Motoichi Ohtsu§, Hirokazu Hori¶, Masahiko Hara†

†Flucto-order Functions Research Team, RIKEN Advanced Science Institute,
‡Tokyo Tech-RIKEN International School, Tokyo Institute of Technology,
2-1, Hirosawa, Wako, Saitama 351-0198, Japan
£Photonic Network Research Institute, National Institute of Information and Communications Technology,
4-2-1 Nukui-kita, Koganei, Tokyo 184-8795, Japan
§Graduate School of Engineering, The University of Tokyo,
2-11-16 Yayoi, Bunkyo-ku, Tokyo 113-8656, Japan
¶Interdisciplinary Graduate School of Medical and Engineering, University of Yamanashi,
Kofu 400-8511, Japan
Email: masashi.aono@riken.jp

Abstract—We propose a biologically-inspired computing algorithm called "AmoebaSAT" for solving an NP-complete combinatorial optimization problem, the Boolean satisfiability problem (SAT). AmoebaSAT is a hybrid of two dynamics; chaotic oscillatory dynamics for exploring the state space are combined with spatiotemporal control dynamics for bouncing back logically-false state transitions. For the former, we employ the logistic map as a unit for generating chaotic fluctuation. The control principle of the latter that we call "bounceback control" is designed to stabilize a state only when it represents a solution, i.e., a satisfiable assignment. We show that, for some benchmark problem instances, AmoebaSAT finds a solution faster than a well-known algorithm called "WalkSAT", which is considered to be one of the fastest algorithms.

1. Introduction

There has been growing interest in biologically-inspired algorithms for solving computationally demanding problems in a fashion similar to search dynamics of various biological systems such as neural networks, evolutionary processes, ants, and swarms [1]. AmoebaSAT extracts the essence of spatiotemporal oscillatory dynamics of a single-celled amoeboid organism, the true slime mold *Physarum polycephalum*, which is capable of searching for a solution to some optimization problems [2]. When placed under our previously studied spatiotemporal control which applies aversive light stimuli locally and dynamically depending on the shape of the organism, the organism exhibits chaotic oscillatory dynamics and finds a solution to the traveling salesman problem by changing its shape into the optimal one for which the area of the body is maximized and the risk of being illuminated is minimized [3]. Inspired by this scheme, we define AmoebaSAT as a hybrid of chaotic oscillatory dynamics and spatiotemporal control dynamics.

The SAT is the problem of determining if a given Boolean formula of N variables $x_i \in \{0\ (false), 1\ (true)\}$ ($i \in I = \{1, 2, \cdots, N\}$) is "satisfiable", i.e., there exists at least one particular assignment of true values to the variables such that it makes the entire formula *true*. Many decision problems and optimization problems can be transformed into instances of the SAT. Thus, SAT solvers are potentially applied to a wide range of practical purposes such as software and hardware design, planning, constraint optimization, and automatic inference.

The SAT is a basis of hard computational problems, as it is the first problem shown to be NP-complete [4]. No algorithm is known to solve the NP-complete problem in a practically tractable time. Indeed, the number of possible assignments 2^N grows exponentially as a function of N.

A "conjunctive normal form (CNF)" of a Boolean formula is the *AND* ($\wedge$) of a series of *clauses*, where each *clause* is the *OR* ($\vee$) of *literals*, and each *literal* is either a variable x_i or its negation $\neg x_i$. The SAT is called "k-SAT" when the formula is a CNF whose *clause* contains at most k variables. 3-SAT is NP-complete whereas 2-SAT is not.

In this study, we formulate AmoebaSAT so that it can be applied to k-SAT. For 20-variable 3-SAT benchmark instances that are available online [6], we compare the performance of AmoebaSAT with that of a widely-studied randomized algorithm for k-SAT, called WalkSAT [5].

2. Models

2.1. AmoebaSAT

2.1.1. Assignment

Given an N-variable formula, AmoebaSAT is defined as a discrete-time-state dynamical system consisting of $2 \cdot N$ units, each of which is labeled with $(i, v) \in I \times \{0, 1\}$ and is an analogy of a pseudopod-like branch of the amoeboid organism. Let $X_{i,v}(t)$ be a displacement of resources in each

unit at time step t. If $X_{i,v} > 0$, we consider that unit (i, v) has "abundant" resources and the system assigns the value v to the ith variable, i.e., $x_i = v$. Accordingly, a *system state* $X = (X_{1,0}, X_{1,1}, X_{2,0}, X_{2,1}, \cdots, X_{N,0}, X_{N,1})$ is mapped to an *assignment* $x = (x_1, x_2, \cdots, x_N)$ as follows:

$$x_i = \begin{cases} 0 & (\text{if } X_{i,0} > 0 \text{ and } X_{i,1} \leq 0), \\ 1 & (\text{if } X_{i,0} \leq 0 \text{ and } X_{i,1} > 1), \\ \text{undefined} & (\text{otherwise}). \end{cases} \quad (1)$$

Consider a four-variable 3-SAT instance, $(x_1 \vee \neg x_2) \wedge (\neg x_2 \vee x_3 \vee \neg x_4) \wedge (x_1 \vee x_3) \wedge (x_2 \vee \neg x_3) \wedge (x_3 \vee \neg x_4) \vee (\neg x_1 \vee x_4)$ in which a "*solution*" $(x_1, x_2, x_3, x_4) = (1, 1, 1, 1)$ uniquely exists. This formula is represented as a set $F = \{\{1, -2\}, \{-2, 3, -4\}, \{1, 3\}, \{2, -3\}, \{3, -4\}, \{-1, 4\}\}$ by replacing x_i and $\neg x_i$ in the formula with i and $-i$, respectively.

2.1.2. Bounceback control dynamics

To make an entire formula *true*, every *clause* needs to be *true* because all the *clauses* in a CNF are connected with *AND* operators. Now, let us focus on a *clause* $(x_1 \vee \neg x_2)$. To make this *clause* "true", if x_1 is $false$, then x_2 should NOT be *true*. Therefore, we introduce the following operation that we call "bounceback control dynamics": If $X_{1,0}(t) > 0$, resource supply to unit $(2, 1)$ is suppressed by applying inhibitory stimulus at the next step $t+1$ as $S_{2,1}(t+1) = 1$. Likewise, scanning all *clauses*, we determine if the inhibitory stimuli are applied ($S_{i,v} = 1$) or not ($S_{i,v} = 0$). For a set-theoretical form F of the given formula, these bounceback control dynamics are written as follows:

$$S_{i,v}(t+1) = \begin{cases} 1 & (\text{if } (B \ni (P, Q) \text{ such that } Q \ni (i, v)) \\ & \text{and (for all } (j, u) \in P,\ X_{j,u}(t) > 0)), \\ 0 & (\text{otherwise}), \end{cases} \quad (2)$$

where

$$B = Intra \cup Inter \cup Contra \quad (3)$$

is a set of bounceback rules whose element (P, Q) is taken as "if all units in P are resource-abundant at t, then suppress all units in Q at $t + 1$".

Intra prohibits each variable x_i to take two values 0 and 1 at a time, i.e., it maintains intra-variable consistency. For each $i \in I$, we append the following element in *Intra*:

$$Intra \ni (\{(i, v)\}, \{(i, 1 - v)\}). \quad (4)$$

Inter defines inter-variable inhibitory coupling. For each *variable* $i \in I$ in each *clause* $C \in F$, we append the following element in *Inter*:

$$Inter \ni \begin{cases} (P, \{(i, 0)\}) & (\text{if } C \ni i), \\ (P, \{(i, 1)\}) & (\text{if } C \ni -i), \end{cases} \quad (5)$$

where, for each $j \neq i$, P includes the following element:

$$P \ni \begin{cases} (j, 0) & (\text{if } C \ni j), \\ (j, 1) & (\text{if } C \ni -j). \end{cases} \quad (6)$$

Some rules in *Inter* may imply that neither 0 nor 1 can be assigned to a variable. To avoid these contradictions, for each $i \in I$, we build *Contra* by checking *Inter* as follows:

$$\text{If } (P, \{(i, 0)\}) \in Inter \text{ and } (P', \{(i, 1)\}) \in Inter, \\ \text{then } Contra \ni (P \cup P', P \cup P'). \quad (7)$$

Intra, *Inter*, and *Contra* of F are shown in Table 1. These rules are determined in a polynomial time $poly(N \cdot M)$, where M is the number of *clauses*.

Table 1: Bounceback rules determined by F.

B	P (if abundant at t)	Q (suppressed at $t+1$)
Intra	$\{(1,0)\}$	$\{(1,1)\}$
	$\{(1,1)\}$	$\{(1,0)\}$
	$\{(2,0)\}$	$\{(2,1)\}$
	$\{(2,1)\}$	$\{(2,0)\}$
	$\{(3,0)\}$	$\{(3,1)\}$
	$\{(3,1)\}$	$\{(3,0)\}$
	$\{(4,0)\}$	$\{(4,1)\}$
	$\{(4,1)\}$	$\{(4,0)\}$
Inter	$\{(2,1)\}$	$\{(1,0)\}$
	$\{(1,0)\}$	$\{(2,1)\}$
	$\{(3,0),(4,1)\}$	$\{(2,1)\}$
	$\{(2,1),(4,1)\}$	$\{(3,0)\}$
	$\{(2,1),(3,0)\}$	$\{(4,1)\}$
	$\{(3,0)\}$	$\{(1,0)\}$
	$\{(1,0)\}$	$\{(3,0)\}$
	$\{(3,1)\}$	$\{(2,0)\}$
	$\{(2,0)\}$	$\{(3,1)\}$
	$\{(4,1)\}$	$\{(3,0)\}$
	$\{(3,0)\}$	$\{(4,1)\}$
	$\{(4,0)\}$	$\{(1,1)\}$
	$\{(1,1)\}$	$\{(4,0)\}$
Contra	$\{(1,1),(3,0)\}$	$\{(1,1),(3,0)\}$
	$\{(1,0),(2,0)\}$	$\{(1,0),(2,0)\}$
	$\{(1,0),(3,1)\}$	$\{(1,0),(3,1)\}$
	$\{(2,1),(4,0)\}$	$\{(2,1),(4,0)\}$
	$\{(2,0),(4,1)\}$	$\{(2,0),(4,1)\}$
	$\{(3,0),(4,0)\}$	$\{(3,0),(4,0)\}$
	$\{(1,1),(2,1),(3,0)\}$	$\{(1,1),(2,1),(3,0)\}$
	$\{(2,0),(2,1),(4,1)\}$	$\{(2,0),(2,1),(4,1)\}$
	$\{(3,0),(3,1),(4,1)\}$	$\{(3,0),(3,1),(4,1)\}$

2.1.3. Chaotic oscillatory dynamics

The dynamics of the units are given as follows:

$$X_{i,v}(t+1) = \begin{cases} X_{i,v}(t) + 1 & (\text{if } (R_{i,v}(t) = 1 \text{ and } |X_{i,v}(t)| < 2) \\ & \text{or } X_{i,v}(t) \leq -2\ ), \\ X_{i,v}(t) - 1 & (\text{if } (R_{i,v}(t) = 0 \text{ and } |X_{i,v}(t)| < 2) \\ & \text{or } 2 \leq X_{i,v}(t)\ ), \end{cases} \quad (8)$$

where $R_{i,v} \in \{0, 1\}$ represents "resource supply" and $X_{i,v} \in \{-2, -1, 0, 1, 2\}$. The following function of the external

stimulus $S_{i,v}$ and internal fluctuation $f_{i,v}$ determines if the resource is supplied ($R_{i,v} = 1$) or bounced back ($R_{i,v} = 0$) :

$$R_{i,v}(t) = \begin{cases} sgn(f_{i,v}(t) - 1 + \beta_+) & (\text{if } S_{i,v}(t) = 1), \\ sgn(1 - f_{i,v}(t) - \beta_-) & (\text{if } S_{i,v}(t) = 0), \end{cases} \quad (9)$$

where $sgn(r) = 1$ if $r > 0$, otherwise 0, and β_+ and β_- are parameters for adjusting the occurrences of errors in stimulus response. Namely, the larger the β_+, the more the resource is likely to be supplied, even though it should be suppressed when $S_{i,v} = 1$. The larger the β_-, the less the resource is likely to supplied, despite the absence of the inhibitory stimulus ($S_{i,v} = 0$). We fix these parameters at $\beta_+ = 0$ and $\beta_- = 0.25$ in this study, because these values were confirmed to be the optimal. The internal fluctuation $f_{i,v}$ is generated by the following logistic map

$$f_{i,v}(t+1) = 4 \cdot f_{i,v}(t) \cdot (1 - f_{i,v}(t)), \quad (10)$$

which produces chaotic behavior.

2.1.4. *Solution search process*

The pseudocode for AmoebaSAT is given as follows.

```
INPUT:    A k-SAT formula F.
OUTPUT:   A satisfiable assignment or "not found"
BEGIN:    Determine B (bounceback rules) of F
          according to Eqs. (3), (4), (5), (6) and (7);
          FOR (i, v) = (1, 0) to (N, 1)
          Set X_{i,v} = 0, R_{i,v} = 0, and S_{i,v} = 0;
          Choose f_{i,v} ∈ [0.0, 1.0] randomly; END FOR
          REPEAT:
          Obtain an assignment x from X by Eq. (1);
          IF x satisfies F THEN RETURN x;
          ELSE      FOR (i, v) = (1, 0) to (N, 1)
                    Update X_{i,v}, R_{i,v}, f_{i,v}, and S_{i,v}
                    according to Eqs. (8), (9), (10),
                    and (2), respectively; END FOR
          END IF
          UNTIL:    Run out of time;
          RETURN:   "not found";
END:
```

A *system state* X is "stabilizable" if the following condition holds for all units (i, v): If $X_{i,v}(t) > 0$ then $S_{i,v}(t) = 0$, or if $X_{i,v}(t) \leq 0$ then $S_{i,v}(t) = 1$. This is because, in a unit where the above condition is not met, in response to $S_{i,v}(t)$, $sgn(X_{i,v}(t+1))$ may differ from $sgn(X_{i,v}(t))$, thus X is unstable. We designed the bounceback control dynamics so that only satisfiable *assignments* (*solutions*) can be stabilizable. In fact, among all possible states, we can confirm that only the states mapped to the solution are stabilizable.

In the example of the solution search process shown in Fig.1, we can confirm that the solution was found after 41 iterations and was likely to be stabilized, where the black arrow indicates the time step when first the inverted binary sequences of $S_{i,v}$ matched with the solution.

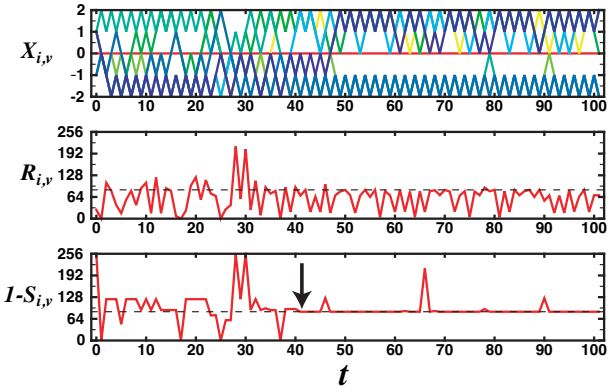

Figure 1: Time evolution of AmoebaSAT solving the four-variable instance given in the text. For $X_{i,v}$, the time series data of 8 units were shown in different colors. For $R_{i,v}$, the binary sequences of 8 units were transformed into a decimal number series, where the solution $(x_1, x_2, x_3, x_4) = (1, 1, 1, 1)$ mapped to 85 was indicated by the black broken line. For $S_{i,v}$, the binary sequences were first inverted (i.e., $S_{i,v} \mapsto 1 - S_{i,v}$), then expressed in a decimal number.

2.2. WalkSAT

WalkSAT is a stochastic local search algorithm which finds a solution with a reasonably large probability after taking a fairly small number of iterations [5].

```
INPUT:    A k-SAT formula F
OUTPUT:   A satisfiable assignment or "not found"
BEGIN:    Choose an assignment x randomly;
          REPEAT:
          IF A satisfies F THEN RETURN x;
          ELSE      Choose a clause C randomly
                    from among unsatisfied clauses;
                    Choose a variable x_i randomly
                    from among C's variables;
                    Update x by flipping x_i;
          END IF
          UNTIL:    Run out of time;
          RETURN:   "not found";
END:
```

In [5], the average number of iterations required for finding a solution was estimated as an exponential function $(2(k-1)/k)^N poly(N)$. Thus, for 3-SAT, WalkSAT requires an average of $(4/3)^N poly(N)$ iterations and is one of the fastest algorithms.

3. Results

The SATLIB website provides benchmark instances [6]. We used a test set of "Uniform Random-3-SAT", which is a family of 3-SAT instance distributions obtained by randomly generating 3-literal CNF formulae. The test set

"uf20-91" contains 1000 instances, each of which is a 20-*variable*-91-*clause* formula. In this study, we focused on 54 instances in the test set, each of which has 8 solutions.

We evaluated the performances of AmoebaSAT and WalkSAT by counting the number of iterations that each algorithm required for finding a solution. For each instance, we performed 500 trials of Monte Carlo simulations.

As shown in Fig. 2, for all instances, AmoebaSAT finds a solution faster than WalkSAT. The average numbers of iterations of the former and latter were approximately 374 and 1173, respectively. The performance of WalkSAT varies wildly depending on instances. Also, it fluctuates greatly depending on trials, as indicated by error bars. In contrast, AmoebaSAT is robust in exhibiting its high performance.

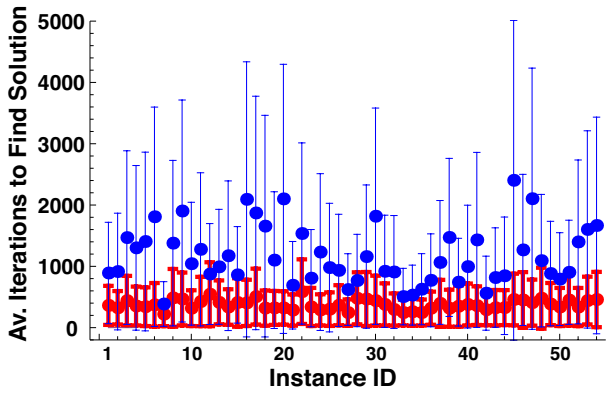

Figure 2: Performance comparison between AmoebaSAT (red) and WalkSAT (blue). For each of 54 benchmark instances, the number of iterations required for finding a solution was averaged over 500 trials, where error bar indicates standard deviation.

4. Discussion and Conclusion

In this study, we demonstrated that AmoebaSAT exhibits its powerful search ability for solving the SAT in a concurrent fashion. AmoebaSAT is a hybrid of chaotic oscillatory dynamics and spatiotemporal control dynamics. We evaluated a modified version of AmoebaSAT in which the chaotic fluctuation Eq. (10) was replaced with random fluctuation (uncorrelated white noise). Compared with the original version, the performance of the modified version degraded significantly, which was even worse than that of WalkSAT. In some previous studies [7, 8], the usefulness of chaotic dynamics for optimization has already been demonstrated with chaotic neural network models and local search algorithms. Why are chaotic fluctuations more powerful than random fluctuations? Some authors reported that negative temporal correlations in chaotic dynamics produce the powerful search abilities [9, 10]. We will examine whether the same applies to AmoebaSAT.

We emphasize that AmoebaSAT is designed to run on some physical substrates that are capable of implementing similar spatiotemporal dynamics, for example, optical energy transfer dynamics between quantum dots mediated by optical near-field interactions [11]. When implemented using these dynamics with spatial and temporal correlations, AmoebaSAT will exhibit its maximum power for larger-sized instances owing to its highly concurrent nature.

References

[1] R. Poli, J. Kennedy, T. Blackwell, "Particle swarm optimization. An overview," *Swarm Intelligence*, vol.1(1), pp.33–57, 2007.

[2] M. Aono, M. Hara, K. Aihara, "Amoeba-based neurocomputing with chaotic dynamics," *Communications of the ACM*, vol.50(9), pp.69–72, 2007.

[3] M. Aono, Y. Hirata, M. Hara, K. Aihara, "Amoeba-based chaotic neurocomputing: Combinatorial optimization by coupled biological oscillators," *New Generation Computing*, vol.27, pp.129–157, 2009.

[4] M. R. Garey, D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," W. H. Freeman and co., New York, 1979.

[5] U. Schoning, "A probabilistic algorithm for *k*-SAT and constraint satisfaction problems," *Proc. 40th Symposium on Foundations of Computer Science*, pp.410–414, 1999.

[6] H. H. Hoos, T. Stutzle, "SATLIB: An online resource for research on SAT," *Proc. SAT2000*, pp.283-292, IOS Press, 2000. Benchmark instances are available online at http://www.cs.ubc.ca/ hoos/SATLIB/benchm.html

[7] K. Aihara, T. Takabe, M. Toyoda, "Chaotic neural networks," *Phys. Lett. A*, vol.144, pp.333–340, 1990.

[8] M. Hasegawa, T. Ikeguchi, K. Aihara, "Combination of chaotic neurodynamics with the 2-opt algorithm to solve traveling salesman problems," *Phys. Rev. Lett.*, vol.79(12), pp.2344–2347, 1997.

[9] M. Hasegawa, K. Umeno, "Solvable performance of optimization neural networks with chaotic noises and stochastic noise with negative correlation," *Lecture Notes in Computer Science*, vol.4984, pp.693–702, Springer, 2008.

[10] K. Umeno, "Performance of chaotic Monte Carlo computation and chaos codes for communications: Theory and experiments," *AIP Conf. Proc.*, vol.1339, pp.197–209, 2011.

[11] M. Naruse, M. Aono, S. -J. Kim, T. Kawazoe, W. Nomura, H. Hori, M. Hara, M. Ohtsu, "Spatiotemporal dynamics in optical energy transfer on the nanoscale and its application to constraint satisfaction problems," submitted.