Stochastic Local Search Algorithms for Constraint Satisfaction*

Silvia Butti[1] and Victor Dalmau[2]

[1] Doctoral student, Universitat Pompeu Fabra, Barcelona, Spain
silvia.butti@upf.edu
[2] PhD advisor, Universitat Pompeu Fabra, Barcelona, Spain
victor.dalmau@upf.edu

Abstract. Stochastic Local Search is a very common tool for finding exact and approximate solutions to Boolean satisfiability problems. While these algorithms work well for 2-SAT, the worst-case runtime is exponential on k-SAT and general non-Boolean CSPs. In this paper we propose an adaptation of a well-known local search algorithm that solves the search problem for the tractable class defined by Cooper et al. [4] in quadratic time.

Keywords: Constraint satisfaction problems · Stochastic local search · Computational complexity.

1 Introduction

A widely used technique in constraint satisfaction is local search. A local search algorithm aims to find a solution to the CSP instance by iteratively moving through a search space consisting of candidate solutions, until either a feasible solution is found, or a time limit elapses. A single step of the algorithm consists of making a local change in the current state which leads to a new state that is close to the previous one in the search space according to some specified measure.

A classical result by Papadimitriou [7] states that there exists a randomised local search algorithm for 2-SAT, which finds a solution, if one exists, in $\mathcal{O}(n^2)$ time with high probability. This algorithm has an immediate adaptation to the Boolean Binary CSP setting, which is known to be polynomial-time tractable.

On the other hand, the general Constraint Satisfaction Problem is NP-complete, and therefore we do not expect there to be a correspondent of Papadimitriou's algorithm for general CSPs that works in polynomial time. An adaptation of the algorithm to arbitrary arity and domain size known as Walk-SAT has indeed been shown to have exponential time complexity [8]. Recently,

* The project that gave rise to these results received the support of a fellowship from "la Caixa" Foundation (ID 100010434). The fellowship code is LCF/BQ/DI18/11660056. This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 713673. Victor Dalmau was supported by a MICCIN grant TIN2016-76573-C2-1P.

it was shown that this algorithm solves k-SAT in polynomial time if the ratio between number of clauses and number of variables is smaller than $\rho\frac{2^k}{k}$ for some small positive constant ρ [3]. A deterministic local search algorithm for k-SAT has been shown to have a similar worst-case upper bound to WalkSAT [5]. Chapedlaine and Creignou [1] showed that the optimization version of Boolean Satisfiability (Weighted Max-SAT) is either in the complexity class P or it is PLS-complete, depending on the constraint language.

However, local search algorithms aimed at solving specific classes of CSPs on arbitrary domains that are known to be tractable did not receive a lot of attention. This is the starting point for the line of research that we propose. The goal of this project is to classify binary constraint satisfaction languages according to whether they are polynomial-time tractable by a local search algorithm. In this paper we present the positive results, namely, we propose a polynomial time stochastic local search algorithm for constraint satisfaction problems, and we show that it solves certain constraint languages with high probability. The proposed algorithm is not only remarkably simple and elegant both in its execution and analysis, but it also improves on the cubic upper bound of the algorithm given in [4] for the same class of languages.

2 Preliminaries

2.1 Constraint Satisfaction Problems

Let D be a finite domain. An instance I of the *Constraint Satisfaction Problem* (CSP) is a triple (V, D, C) where V is a set of variables and C is a set of constraints over the variables. A constraint $c \in C$ is a pair $(\mathbf{v}, R)$ where $R \subseteq D^k$ is a relation over D of finite arity k, and $\mathbf{v}$ is a tuple of k variables from V, known as the *scope* of c. An *assignment* $s : V \to D$ is said to be *satisfying* if for all constraints $c = (\mathbf{v}, R) \in C$ we have $s(\mathbf{v}) \in R$, where s is applied to $\mathbf{v}$ component-wise. We denote the number of variables by n.

Let Γ be a finite set of relations over D, and let CSP(Γ) denote the set of CSP instances with all constraint relations lying in Γ. In this context, Γ is known as the *constraint language*. Then, the *decision problem* for CSP(Γ) is the problem of deciding whether a satisfying assignment exists for an instance $I \in$ CSP(Γ). The *search problem* for CSP(Γ) is the problem of finding a satisfying assignment, given a satisfiable instance $I \in$ CSP(Γ). It is well known that the decision and search problems for CSP are equivalent [2]. In particular then, if no polynomial time algorithm exists to solve the search problem for some constraint language Γ, the same is true for the decision problem on Γ.

2.2 0/1/All Constraints

In 1994, Cooper et al. [4] investigated a class of binary relations that, due to the particular symmetry of their structure, gives rise to tractable problems. Let R be a binary relation over D and let $A, B \subset D$ be the smallest subsets of D such

that R is contained in $A \times B$. In other words, A and B are the projections of R to the first and second coordinate respectively. We say that two elements $a \in A$, $b \in B$ are R-consistent if $(a, b) \in R$. The relation R is said to be *0/1/all* (or ZOA for short) if every element in A is R-consistent with exactly zero, one, or all elements of B, and vice versa every element in B is R-consistent with exactly zero, one, or all elements of A. Cooper et al. also showed that every ZOA relation can be characterised to be one of three types: *permutation*, *2-fan*, or *complete*. A binary relation $R \subseteq D^2$ is a *permutation relation* if $R = \{(x, \tau(x)\}_{x \in A}$ for some bijection $\tau : A \to B$. The name derives from the fact that if $A = B$, then τ is a permutation. A binary relation $R \subseteq D^2$ is a *2-fan relation* if there exist $x \in A$, $y \in B$ such that $R = (\{x\} \times B) \cup (A \times \{y\})$. A binary relation $R \subseteq D^2$ is a *complete relation* if $R = A \times B$. Likewise, $c = ((u, v), R) \in C$ is a permutation (2-fan, complete) constraint if R is a permutation (2-fan, complete) relation respectively. Examples of such relations are shown in Figure 1.

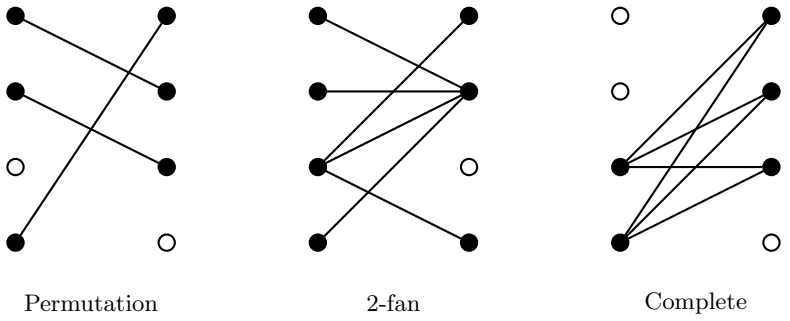

Fig. 1. Examples of permutation, 2-fan, and complete relations. The elements of A, B are shown in black and the elements of $D \setminus A$, $D \setminus B$ are shown in white.

2.3 Local Search Algorithms

The following is our working characterisation of a local search algorithm. For an overview of SLS algorithms and their applications, we refer the reader to [6].

Definition 1. *Given an instance I of a combinatorial decision problem, a randomised local search algorithm for I is defined by:*

- *A finite set of states or candidate solutions $S(I)$, known as the search space;*
- *A set of feasible solutions $S_f(I) \subseteq S(I)$;*
- *A set of starting states $S_0(I) \subseteq S(I)$;*
- *A binary neighbourhood relation $N(I) \subseteq S(I)^2$ defined on the search space;*
- *A step function $step(I) : S(I) \to \mathcal{D}(S(I))$ mapping each search position onto a probability distribution over its neighbouring search positions.*

The SLS algorithm terminates either when a feasible solution $s \in S_f(I)$ has been found, or when a pre-set time limit has passed.

Note that the step function depends exclusively on the current state of the algorithm: no memory of the previous states is needed, hence why these kind of algorithms are said to be *memoryless*. Moreover, we allow for stochasticity, that is, the steps of the algorithm are defined by their probability distribution. Note that this is a generalisation with respect to deterministic algorithms, and therefore allows for determinism too when necessary. Finally, we point out that in our positive results, the initial state s_0 is irrelevant, meaning that the algorithm that we propose is guaranteed to converge given any starting configuration. We say that a SLS algorithm solves $\mathrm{CSP}(\Gamma)$ *with high probability* if it finds a feasible solution $s \in S_f(I)$ with some probability p such that $\lim_{k\to\infty} p = 1$ for some parameter k which can be made arbitrarily large.

In our specific case, the set $S(I)$ of candidate solutions corresponds to the set of maps $s : V \to D$, while the set of feasible solutions corresponds to the set of satisfying assignments for I. We always assume that $S_f(I)$ is nonempty. The starting state can be any of the candidate solutions. The neighbourhood relation $N(I)$ contains all the pairs of candidate solutions that differ in the value assigned to exactly one variable. We will see that the step function can take a very easy characterization. Specifically, rather than defining it on every element of $S(I)$, we will only need to define a probability function on $D^k \setminus R$ for every k-ary relation R in the constraint language of I. With a simple trick, this function induces a probability function on the entire candidate solution space.

3 Results

This section is entirely dedicated to proving the main result of this paper, which we outline in Theorem 1.

Theorem 1. *Let $I = (V, D, C) \in \mathrm{CSP}(\Gamma)$ be an instance with $|V| = n$. If all the relations in Γ are binary ZOA, then there exists a randomised local search algorithm which finds a satisfying assignment, if one exists, in $\mathcal{O}(|D|n^2)$ time with high probability.*

Note that this is an improvement of a factor of n with respect to the algorithm proposed in [4], which depends on both the number of edges and the number of variables in a CSP instance and hence has worst-case runtime cubic in n.

Now, Papadimitriou's algorithm for 2-SAT is effectively a quadratic SLS algorithm which solves all binary CSPs on domains of size 2. Therefore, we only need to show that Theorem 1 holds for ZOA relations with domain of size at least 3.

3.1 The SLS Algorithm for ZOA constraints

In Algorithm 1 we present the SLS procedure that solves with high probability all CSP instances whose constraints are all ZOA. Note that while the procedure

is the same, the step function differs among the three types of ZOA relations. We outline the specific functions below.

Algorithm 1: Randomised Local Search for ZOA Constraints

```
Result: A satisfying assignment s : V → D
for run ← 1 to k do
    Pick an arbitrary initial assignment s_0 from S_0(I)
    t ← 1
    while t ≤ T do
        Pick an arbitrary constraint c = ((u, v), R) ∈ C that is currently
          unsatisfied
        Draw (s_t(u), s_t(v)) from the probability distribution given by
          step(s_{t−1}(u), s_{t−1}(v))
        Let s_t(w) = s_{t−1}(w) for all w ∈ V \ {u, v}
        if s_t is a satisfying assignment then
            return s_t
            break
        end
        t ← t + 1
    end
    run ← run + 1
end
```

Let $R \subseteq D^2$ be a ZOA relation and let $A, B \subseteq D$ be the smallest sets such that $R \subseteq A \times B$. Let s_t be the current assignment and the constraint $c = ((u, v), R)$ be such that $s_t(u, v) \notin R$. Let $p = \frac{1}{n}$, where n is the size of the CSP instance.

Permutation constraints. If R is a permutation relation with $\tau : A \to B$ a bijection such that $R = \{(d, \tau(d))\}_{d \in A}$, then

- If $s_t(u, v) \notin A \times B$, use the probability distribution function given for complete constraints.
- If $s_t(u, v) = (a, b)$ for some $a \in A$, $b \in B$, then
  - $s_{t+1}(u, v) = (a, \tau(a))$ with probability $\frac{1}{2}(1 - p)$
  - $s_{t+1}(u, v) = (\tau^{-1}(b), b)$ with probability $\frac{1}{2}(1 - p)$
  - $s_{t+1}(u, v) = (a, b')$ with probability $\frac{1}{2}p$, where b' is chosen uniformly at random from $B \setminus \{b, \tau(a)\}$
  - $s_{t+1}(u, v) = (a', b)$ with probability $\frac{1}{2}p$, where a' is chosen uniformly at random from $A \setminus \{a, \tau^{-1}(b)\}$

2-fan constraints. If R is a 2-fan relation and $x \in A$, $y \in B$ are such that $R = (\{x\} \times B) \cup (A \times \{y\})$, then

- If $s_t(u, v) = (x, d)$ for some $d \in D \setminus B$
  - $s_{t+1}(u, v) = (x, y)$ with probability p

  - $s_{t+1}(u,v) = (x,b)$ with probability $1-p$, where b is chosen uniformly at random from $B \setminus \{y\}$
- If $s_t(u,v) = (d,y)$ for some $d \in D \setminus A$,
  - $s_{t+1}(u,v) = (x,y)$ with probability p
  - $s_{t+1}(u,v) = (a,y)$ with probability $1-p$, where a is chosen uniformly at random from $A \setminus \{x\}$
- If $s_t(u,v) = (d,d')$ for some $d \in D \setminus \{x\}$ and $d' \in D \setminus \{y\}$
  - $s_{t+1}(u,v) = (x,d')$ with probability $\frac{1}{2}$
  - $s_{t+1}(u,v) = (d,y)$ with probability $\frac{1}{2}$

Complete constraints. If R is a complete relation with $R = A \times B$, then

- If $s_t(u,v) = (a,d)$ for some $d \in D \setminus B$, then $s_{t+1}(u,v) = (a,b)$ where b is chosen uniformly at random from B
- If $s_t(u,v) = (d,b)$ for some $d \in D \setminus A$, then $s_{t+1}(u,v) = (a,b)$ where a is chosen uniformly at random from A
- If $s_t(u,v) = (d,d')$ for some $d \in D \setminus A$ and $d' \in D \setminus B$
  - $s_{t+1}(u,v) = (a,d')$ with probability $\frac{1}{2}$, where a is chosen uniformly at random from A
  - $s_{t+1}(u,v) = (d,b)$ with probability $\frac{1}{2}$, where b is chosen uniformly at random from B

Now consider a constraint language Γ which contains exclusively ZOA relations. We are going to show that the SLS algorithm which applies each of the above versions of the step function to the relevant type of constraint - permutation, 2-fan, or complete - solves the search problem for any instance of CSP(Γ) in polynomial time with high probability.

Lemma 1. *Let $I = (V,D,C) \in \mathrm{CSP}(\Gamma)$ be an instance with $|V| = n$ and $|D| \geq 3$. If all the relations in Γ are binary ZOA, then Algorithm 1 finds a satisfying assignment, if one exists, in $\mathcal{O}(|D|n^2)$ time with high probability.*

Proof. Let $I = (V,D,C)$ be a CSP instance as in the premises of Lemma 1. Let $s_t : V \to D$ be the current assignment given by Algorithm 1 at time t. Assume that I is satisfiable, and let $\sigma : V \to D$ be a satisfying assignment of I. Let X_t be a random variable denoting the number of variables in V on which s_t agrees with the target assignment σ. The algorithm terminates when $X_t = n$, or when $t > T$.[3] Denote by $q_{i,k}$ the probability $\mathbb{P}[X_{t+1} = i+k | X_t = i]$ and notice that due to the memoryless nature of our SLS algorithm, the sequence of random variables X_t forms a Markov chain. Clearly $q_{i,k} = 0$ unless $|k| \leq 1$, so we only need to consider the values of k in $\{-1,0,1\}$.

Consider an arbitrary unsatisfied constraint $c = ((u,v),R)$. Notice that this can be picked arbitrarily, and so it can also be chosen in an adversarial fashion. Define

$$C'_t := \{c = ((u,v),R) \in C : (s_t(u) = \sigma(u)) \vee (s_t(v) = \sigma(v))\}.$$

[3] Note that it may terminate sooner if it reaches a different satisfying assignment, but it is not necessary to consider this in the analysis.

We must divide our analysis into two separate cases depending on whether or not $c \in C'_t$. The values of $q_{i,k}$ depend on which version of the step function we apply to c - which in turn depends on whether R is a permutation, 2-fan, or complete relation. Due to space restrictions, we only report the relevant values of $q_{i,k}$ (see Table 1) and the proof of the exact time bounds for permutation constraints. A similar case analysis yields the values of $q_{i,k}$ and hence the time bounds for the other types of constraints.

Table 1. Values of $q_{i,k}$ for permutation constraints, where c is the unsatisfied constraint picked at time t. Without loss of generality, we are assuming that $s_t(u,v) \in A \times B$ where (u,v) is the scope of c, since the algorithm for $s_t(u,v) \notin A \times B$ adds at most a linear factor to the running time. Notice that for permutation relations, $|A| = |B| \leq |D|$.

	$c \in C'_t$	$c \notin C'_t$
$q_{i,+1}$	$\frac{1}{2}(1-p)$	$\frac{p}{\lvert A\rvert-2}$
$q_{i,0}$	$\frac{1}{2}p$	$1 - \frac{p}{\lvert A\rvert-2}$
$q_{i,-1}$	$\frac{1}{2}$	0
Indices	$i = 1, \ldots, n-1$	$i = 0, \ldots, n-2$

Let h_j be the expected number of time steps needed for assignment s_t to agree with assignment σ on all variables, given that s_0 starts $n-j$ variables away from σ (that is, at time $t = 0$, s_t and σ agree on exactly j variables). Then, $h_j = q_{i,+1}h_{j+1} + q_{i,0}h_j + q_{i,-1}h_{j-1} + 1$. Notice that, when $X_t = 0$, the set C'_t is empty. Therefore, we have a reflecting barrier at

$$h_0 = h_1 + \frac{|A| - 2}{p} \tag{1}$$

If $X_t = n-1$, then all unsatisfied constraints are in C'_t. In all other cases, the constraint that will be probed can be either in C'_t or in $C \setminus C'_t$.

Our goal is to find a polynomial upper bound for h_j. We claim that, in all cases,

$$h_j \leq h_{j+1} + \frac{1}{(1-p)^j}\frac{|D| - 2}{p} + 2\sum_{i=1}^{j}\frac{1}{(1-p)^i}. \tag{2}$$

The proof is by induction. The case $j = 1$ is given by (1). Now suppose the claim holds for all $k < j$. If the constraint chosen at time t is in $C \setminus C'_t$, it is easy to see that

$$h_j = h_{j+1} + \frac{|A| - 2}{p} \leq h_{j+1} + \frac{1}{(1-p)^j}\frac{|D| - 2}{p} + 2\sum_{i=1}^{j}\frac{1}{(1-p)^i}. \tag{3}$$

Else, if the chosen constraint is in C'_t, using the induction hypothesis we get exactly the inequality (2), so in either cases, our claim holds. Now we have that

$\frac{1}{(1-p)^i} \leq (\frac{n}{n-1})^n$, therefore

$$\sum_{i=1}^{j} \frac{1}{(1-p)^i} \leq n\Big(1+\frac{1}{n-1}\Big)^n - n \leq ne^{\frac{n}{n-1}} - n \longrightarrow (e-1)n \tag{4}$$

and

$$h_j \leq h_{j+1} + \Big(\frac{n}{n-1}\Big)^n n(|D|-2) + 2(e-1)n \longrightarrow h_{j+1} + n(e|D|-2). \tag{5}$$

From a similar analysis we deduce that for 2-fan constraint, the bound on h_j is $h_j \leq h_{j+1}+2j+n$ and for complete constraints, we have $h_j \leq h_{j+1}+|D|$. Then, by combining these values we get a unique upper bound for all ZOA constraints:

$$h_j \leq h_{j+1} + 2j + n(|D|e-2). \tag{6}$$

Then, the absorbing barrier at $h_n = 0$ implies that

$$h_j \leq (n-j)(n+j+1+n(e|D|-2) \leq e|D|n^2 \tag{7}$$

Choose $T = \lceil 2e|D|n^2 \rceil$ and let T_j be the number of time steps required to reach $X_{T_j} = n$ in a run of the algorithm with $X_0 = j$. Note that $\mathbb{E}[T_j] = h_j$. Then, by Markov's inequality, the probability that the algorithm succeeds is

$$1 - \mathbb{P}[T_j \geq \lceil 2e|D|n^2 \rceil] \geq 1 - \frac{h_j}{\lceil 2e|D|n^2 \rceil} \geq 1 - \frac{e|D|n^2}{2e|D|n^2} = \frac{1}{2}.$$

So for k runs of the algorithm, the probability that it succeeds at least once is at least $1 - 2^{-k}$, as required. □

3.2 Future work

Cooper et al. showed that for any Γ which is closed under permutations and composition with unary relation, CSP(Γ) is polynomial time tractable if and only if every element of Γ is a ZOA relation. We ask whether the same holds if we restrict our power to SLS algorithms only. That is, we conjecture that Theorem 1 is almost a necessary condition, therefore, we hope to prove that all the constraint languages that are not ZOA and do not fall in a small class of trivial languages are in fact intractable by SLS. The next obvious open question is to define the border of tractability by SLS for non-binary constraints.

References

1. Chapdelaine, P., Creignou, N.: The complexity of boolean constraint satisfaction local search problems. Annals of Mathematics and Artificial Intelligence **43**(1-4), 51–63 (2005)
2. Cohen, D.A.: Tractable decision for a constraint language implies tractable search. Constraints **9**(3), 219–229 (2004)

3. Coja-Oghlan, A., Frieze, A.: Analyzing walksat on random formulas. In: Proceedings of the Meeting on Analytic Algorithmics and Combinatorics. p. 48–55. ANALCO '12, Society for Industrial and Applied Mathematics, USA (2012)
4. Cooper, M.C., Cohen, D.A., Jeavons, P.G.: Characterising tractable constraints. Artificial Intelligence **65**(2), 347–361 (1994)
5. Dantsin, E., Goerdt, A., Hirsch, E.A., Kannan, R., Kleinberg, J., Papadimitriou, C., Raghavan, P., Schöning, U.: A deterministic $(2-2/(k+1))^n$ algorithm for k-sat based on local search. Theoretical Computer Science **289**(1), 69 – 83 (2002)
6. Hoos, H., Sttzle, T.: Stochastic Local Search: Foundations & Applications. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2004)
7. Papadimitriou, C.H.: On selecting a satisfying truth assignment. [1991] Proceedings 32nd Annual Symposium of Foundations of Computer Science pp. 163–169 (1991)
8. Schoning, T.: A probabilistic algorithm for k-sat and constraint satisfaction problems. In: 40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039). pp. 410–414. IEEE (1999)