Variable Neighborhood Search for the Probabilistic Satisfiability Problem

Dejan Jovanović* Nenad Mladenović† Zoran Ognjanović*

*Mathematical Institute
Kneza Mihaila 35, 11000 Belgrade, Serbia and Montenegro
{dejan,zorano}@mi.sanu.ac.yu

†School of Mathematics, University of Birmingham
Edgbaston, Birmingham, United Kingdom
N.Mladenovic@bham.ac.uk

1 Introduction

In the field of artificial intelligence researchers have studied uncertain reasoning using different tools. Some of the formalisms for representing and reasoning with uncertain knowledge are based on probabilistic logic [1, 9]. This logic extends classical propositional language with expressions that speak about probability, while formulas remain true or false. The satisfiability problem of a probabilistic formula (PSAT) is NP-complete [1]. Although PSAT can be reduced to linear programming problem, solving it by any standard linear system solving procedure is unsuitable in practice, due to exponential growth of number of variables in the linear system obtained by the reduction. Nevertheless, it is still possible to use more efficient numerical methods, e.g., the column generation procedure of linear programming. In more recent article [8] we presented Genetic algorithms (GA) for PSAT. In this paper we suggest Variable neighborhood search (VNS) based heuristic for solving PSAT.

2 Probabilistic Logic and PSAT

In this section we introduce the PSAT problem formally (for a more detailed description please see [9]). Let $\mathcal{L} = \{x, y, z, \ldots\}$ be the set of propositional letters. A *weight term* is an expression of the form $a_1 w(\alpha_1) + \ldots + a_n w(\alpha_n)$, where a_i's are rational numbers, and α_i's are classical propositional formulas with propositional letters from $\mathcal{L}$. The intended meaning of $w(\alpha)$ is probability of α being true. A *basic weight formula* has the form $t \geq b$, where t is a weight term, and b is a rational number. Formula $(t < b)$ denotes $\neg(t \geq b)$. A *weight literal* is an expression of the form $t \geq b$ or $t < b$. The set of all weight formulas is the minimal set that contains all basic weight formulas, and it is closed under Boolean operations. Let α be a classical propositional formula and $\{x_1, x_2, \ldots, x_k\}$ be the set of all propositional letters that

Repeat the following steps until the stopping condition is met:

1. Set $k \leftarrow 1$;
2. Until $k = k_{\max}$, repeat the following steps:
   (a) *Shaking.* Generate a point x' at random from the k^{th} neighborhood of x ($x' \in \mathcal{N}_k(x)$);
   (b) *Local Search.* Apply some local search method with x' as initial solution; denote with x'' the so obtained local optimum;
   (c) *Move or not.* If this optimum is better than the the incumbent, move there ($x \leftarrow x''$), and continue the search with $\mathcal{N}_1(k \leftarrow 1)$; otherwise, set $k \leftarrow k + 1$;

Figure 1: Main steps of the basic VNS metaheuristic

appear in α. An *atom* of α is defined as a formula $at = \pm x_1 \wedge \ldots \wedge \pm x_k$ where $\pm x_i \in \{x_i, \neg x_i\}$. Let $\text{At}(\alpha)$ denote the set $\{at_1, \ldots, at_{2^k}\}$ of all atoms of α. Every classical propositional formula α is equivalent to a formula $\text{CDNF}(\alpha) = \vee_{j=1}^{m} at_{i_j}$ called complete disjunctive normal form of α. We use $at \in \text{CDNF}(\alpha)$ to denote that the atom at appears in $\text{CDNF}(\alpha)$. A formula f is in the *weight conjunctive form* (WCF) if it is a conjunction of weight literals. Every weight formula f can be transformed to a disjunctive normal form

$$\text{DNF}(f) = \vee_{j=1}^{m} \wedge_{i=1}^{k_j} (a_1^{i,j} w(\alpha_1^{i,j}) + \ldots + a_{n_{i,j}}^{i,j} w(\alpha_{n_{i,j}}^{i,j}) \ \rho_{i,j} \ b_{i,j}),$$

where disjuncts are formulas in WCF and each $\rho_{i,j}$ can be either $<$ or $\geq$. Since a disjunction is satisfiable iff at least one disjunct is satisfiable, we will consider only WCF formulas.

Formally, PSAT is the following problem: given a WCF formula f, is there any probability function μ defined on $\text{At}(f)$ such that f is satisfiable? Note that a WCF formula f is satisfied by μ iff for every weight literal $a_1^i w(\alpha_1^i) + \ldots + a_{n_i}^i w(\alpha_{n_i}^i) \ \rho_i \ b_i$ in f

$$a_1^i \sum_{at \in \text{CDNF}(\alpha_1^i)} \mu(at) + \ldots + a_{n_i}^i \sum_{at \in \text{CDNF}(\alpha_{n_i}^i)} \mu(at) \ \rho_i \ b_i, \tag{1}$$

while the probabilities $\mu(at)$ assigned to atoms are nonnegative and sum up to 1. This system contains a row for each weight literal of f, and columns that correspond to atoms $at \in \text{At}(f)$ that belong to $\text{CDNF}(\alpha)$ for at least one propositional formula α from f.

3 VNS for the PSAT

VNS is a recent metaheuristic for solving combinatorial and global optimization problems (e.g, see [6, 3, 4]). It is a simple, yet very effective metaheuristic that has shown to be very robust on a variety of practical NP-hard problems. The basic idea behind VNS is change of neighborhood structures in the search for a better solution. In an initialization phase, a set of $k_{\max}$ (a parameter) neighborhoods is preselected ($\mathcal{N}_i, i = 1, \ldots, k_{\max}$), a stopping condition determined and an initial local solution found. Then the main loop of the method has the steps described in Figure 1. Therefore, to construct different neighborhood structures and to perform a systematic search, one needs to supply the solution space with some metric (or quasi-metric) and then induce neighborhoods from it. In the next sections, we answer this problem-specific question for the PSAT problem.

The Solution Space. As the PSAT problem reduces to a linear program over probabilities, the number of atoms with nonzero probabilities necessary to guarantee that a solution will be found, if one exists, is equal to $L + 1$. Here L is the number of weight literals in the WCF formula. The solution is therefore an array of $L + 1$ atoms $x = [A_1, A_2, \ldots, A_{L+1}]$, where A_i, $i = 1, \ldots, L + 1$ are atoms from $At(f)$, with the assigned probabilities $p = [p_1, p_2, \ldots, p_{L+1}]$. Atoms are represented as bit strings, with the i^{th} bit of the string 1 iff the i^{th} variable is positive in the atom. A solution (an array of atoms) is the bit string obtained by concatenation of the atom bit strings. Observe that if the variable x is known, the probabilities of atoms not in x are 0, and system (1) can be rewritten compactly as

$$\sum_{j=1}^{L+1} c_{ij} p_j \quad \rho_i \quad b_i, \quad i = 1, \ldots, L \ . \tag{2}$$

Here $c_{ij} = \sum a_k^i$, with sum going over all α_k^i from (1) such that $A_j \in \text{CDNF}(\alpha_k^i)$. Solving (2) gives us a vector of probabilities p.

Initial Solution. Initial solution is obtained as follows. First, $10 \times (L + 1)$ atoms are randomly generated, and then from these atoms the $L + 1$ atoms with the best grades are selected to form an initial solution. A *grade of an atom* is computed as the sum of this atom's contribution to satisfiability of the conjuncts in the WCF formula. An atom corresponds to a column of the linear system (2). If one coefficient from the column is positive, and located in a row with $\geq$ as the relational symbol, it can be used to push toward the satisfiability of the row. In such a case we add its value to the grade. The $<$ case is not in favor of satisfying the row, so we subtract the coefficient from the grade. Similar reasoning is applied when the coefficient is negative. After initial atoms are selected as described, they are all assigned equal probabilities.

Neighborhood Structures. The neighborhood structures are those induced by the Hamming distance on the solution bit strings. The distance between two solutions is the number of corresponding bits that differ in their bit string representations. With this selection of neighborhood structures a *shake* in the k^{th} neighborhood of a solution is obtained by inverting k bits in the solution's bit string.

3.1 Finding Probabilities by VND

When we change some atoms there may be a better probabilities assignment in the solution, i.e., one that is closer to satisfying the PSAT problem. At this point, we suggest three procedures within Variable neighborhood descent (VND) framework: two fast heuristics and Nelder Mead nonlinear programming method.

Nonlinear Optimization Approach. To find a possibly better probabilities assignment a nonlinear program is defined with the objective function being the distance of the left side from the right side of the linear system (2). Let x be the current solution, we define an

unconstrained objective function z to be

$$z(p) = \sum_{i=1}^{L} d_i(p) + g(p)$$

where d_i is the distance of the left and the corresponding right hand side of the i^{th} row defined as

$$d_i(p) = \begin{cases} (\sum_{j=1}^{L+1} c_{ij}p_j - b_i)^2 & \text{if the } i^{\text{th}} \text{ row is not satisfied,} \\ 0 & \text{otherwise.} \end{cases}$$

and $g(p)$ is the penalty function

$$g(p) = \mu_1(\sum_{p_i<0} p_i^2) + \mu_2(1 - \sum_{i=1}^{L+1} p_i)^2$$

with penalty parameters μ_1 and μ_2. The penalty function is used to transform the constrained nonlinear problem to the unconstrained one. Since the unconstrained objective function $z(p)$ is not differentiable a minimization method that does not use derivatives is needed. We applied the downhill simplex method of Nelder and Mead [7], however, in order to speed-up the search, this optimization procedure is used only when the VNS algorithm reaches $k_{\max}$.

Heuristic Approach. Nonlinear optimization has shown to be too time demanding, so we resorted to a heuristic approach that is used for the majority of the solution optimizations. The heuristic optimization consists of the following two independent heuristics.

Worst Unsatisfied Projection. The 5 most unsatisfied rows are selected, i.e. the rows with the largest values $d_i(p)$. The equations of these rows define corresponding hyper-planes that border the solution space. In an attempt to reach the solution space, consecutive projections of probabilities on the these hyper-planes are performed. Using normal projection the probabilities are changed toward satisfiability of each worst row in a fastest possible manner, i.e. in the direction normal to that hyper-plane. This procedure is repeated at most 10 times or until no improvement has been achieved, every time selecting the current 5 worst rows.

Greedy Giveaway. Again, the worst unsatisfied row is selected (the i^{th}), but also the most satisfied row is selected (the j^{th}). Nonzero coefficients are found in these two rows, but only those that have zero coefficients at the same position in the other row. For example, if $c_{i,k}$ and $c_{j,l}$ are such coefficients, then p_k can be changed with the the most unsatisfied row moving toward satisfiability, and p_l can be changed the opposite way, reducing the satisfiability of the most satisfied row. This procedure is repeated for all suitable pairs of the coefficients from the two rows.

These two heuristics together (first one, then the other) yield a great improvement of the objective function value. Improvements obtained are comparable to those obtained by nonlinear optimization, but they are much faster. This is the reason why the nonlinear optimization is used only after the $\mathcal{N}_{k_{\max}}$ neighborhood is unsuccessfully explored.

Local Search. The local search part of the algorithm scans through the first neighborhood of a solution in pursue of a solution better than the current best. Note that, since the $\mathcal{N}_1$

neighborhood is huge, it would be very inefficient to recompute the probabilities at every point. Therefore, throughout the local search, the probabilities are are fixed. Solutions are compared only by the value of the objective function z. This value is kept for the solutions that are in use. As for the newly generated solutions, the objective value z is computed using the current probabilities assignment, but since only one bit of one atom is changed, the updating is in $O(L)$. The VNS implementation used is stated in Algorithm 1.

Algorithm 1 Basic VNS for PSAT

```
1: x ← initialSolution(); improve(x, heuristic)
2: while (not done()) do
3:     k ← 1
4:     while (k ≤ k_max) do
5:         x' ← shake(x, k); improve(x', heuristic); x'' ← localSearch(x')
6:         if (x'' better than x) then
7:             x ← x''; improve(x, heuristic); k ← 1
8:         else
9:             k ← k + 1
10:        end if
11:    end while
12:    improve(x, nelder-mead)
13: end while
```

4 Computational Results

For testing purposes a set of 27 random satisfiable probability WCF formulas has been generated. Maximal number of summands in weight terms (S), and the maximal number of disjuncts (D) in DNFs of propositional formulas has been set to 5. Three problems instances were generated for each of the following (N, L) pairs: (50, 100), (50, 250), (100, 100), (100, 200), (100, 500), (200, 200), (200, 400), (200, 1000). The VNS algorithm was run with $k_{\max}$ set to 30, and we exit if a feasible solution is found or the method doesn't advance in 10 consecutive iterations. The results are shown in Table 1. It can be seen that VNS approach outperforms our previous checker based on the GA-approach [8] in almost all problem instances, with both increase in the solving success rate and decrease of the execution time. We are not aware of any larger PSAT problem instances reported in the literature. For example, L is up to 500 in [2], and $N = 200$, $L = 800$ in [5]. Also, the instances considered in mentioned papers have a simpler form than the ones used here, with weight terms containing the probability of only one clause with up to 4 literals.

5 Conclusion

In this paper we suggest VNS based heuristic for solving PSAT problem. This approach allows us to solve large problem instances more effectively and more efficiently than the recent GA based heuristic. There are many directions for further research: (i) improve efficiency of our VNS based heuristic by reducing the large neighborhood or by introducing new ones to be used within local search; (ii) apply our approach to the so-called interval PSAT [2] in which basic weight formulas are of the form $c_1 \leq t \leq c_2$; (iii) try to conjecture some relation between the parameters and the hardness of the PSAT problem instance (similar to the phase transition

Table 1: Computational results of the VNS method compared to the previous GA approach.

	VNS	GA		VNS	GA
N, L, instance	solved (cpu time)	solved (cpu time)	N, L, instance	solved (cpu time)	solved (cpu time)
50, 100, 1	26/30 (29.46)	23/30 (188.07)	100, 500, 1	30/30 (230.97)	21/30 (11369.29)
50, 100, 2	30/30 (0.23)	27/30 (21.98)	100, 500, 2	23/30 (1930.52)	19/30 (18932.76)
50, 100, 3	29/30 (17.72)	13/30 (194.04)	100, 500, 3	2/30 (30460.00)	12/30 (17907.2)
50, 250, 1	30/30 (32.60)	25/30 (504.58)	200, 200, 1	0/30 (–)	0/30 (–)
50, 250, 2	15/30 (14.60)	25/30 (2515.88)	200, 200, 2	30/30 (2.30)	27/30 (253.82)
50, 250, 3	30/30 (11.37)	25/30 (664.7)	200, 200, 3	30/30 (0.27)	28/30 (48.36)
100, 100, 1	30/30 (0.03)	30/30 (6.19)	200, 400, 1	30/30 (9.73)	28/30 (524.63)
100, 100, 2	30/30 (0.13)	30/30 (13.28)	200, 400, 2	12/30 (6082.67)	16/30 (12167.64)
100, 100, 3	30/30 (0.03)	30/30 (66.85)	200, 400, 3	2/30 (5531.50)	15/30 (11938.51)
100, 200, 1	30/30 (4.23)	27/30 (858.85)	200, 1000, 1	30/30 (966.47)	22/30 (68437.78)
100, 200, 2	27/30 (79.07)	10/30 (1395.62)	200, 1000, 2	0/30 (–)	0/30 (–)
100, 200, 3	25/30 (77.04)	10/30 (1235.31)	200, 1000, 3	30/30 (1942.93)	21/30 (56081.48)

phenomenon for SAT). Thus, an exhaustive empirical study should give better insight into this phenomenon.

References

[1] R. Fagin, J. Halpern, and N. Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87(1/2):78–128, 1990.

[2] P. Hansen and B. Jaumard. Probabilistic satisfiability. In *Algorithms for uncertainty and defeasible reasoning*, volume 5 of *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, pages 321–367. Kluwer Academic Publishers, 2001.

[3] P. Hansen and N. Mladenović. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449–467, May 2001.

[4] P. Hansen and N. Mladenović. Variable neighborhood search. In *Handbook of Metaheuristics*, volume 57 of *International Series in Operations Research and Management Science*, pages 145–184. Springer, 2003.

[5] P. Hansen and S. Perron. Merging the local and global approaches to probabilistic satisfiability. Technical report, GERAD, 2004.

[6] N. Mladenović and P. Hansen. Variable neighborhood search. *Computers and Operations Research*, 24(11):1097–1100, 1997.

[7] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[8] Z. Ognjanović, U. Midić, and J. Kratica. A genetic algorithm for probabilistic sat problem. In *Artificial Intelligence and Soft Computing*, volume 3070 of *Lecture Notes in Computer Science*, pages 462–467. Springer, 2004.

[9] Z. Ognjanović and M. Rašković. Some first-order probability logics. *Theoretical Computer Science*, 247(1-2):191–212, 2000.