THE MAXIMUM SATISFIABILITY PROBLEM, *MAX-SAT*

In the Maximum Satisfiability (*MAX–SAT*) problem one is given a Boolean formula in conjunctive normal form, i.e., as a conjunction of clauses, each clause being a disjunction. The task is to find an assignment of truth values to the variables that satisfies the maximum number of clauses.

Let n be the number of variables and m the number of clauses, so that a formula has the following form:

$$\bigwedge_{1\le i\le m} \left(\bigvee_{1\le k\le |C_i|} l_{ik} \right)$$

where $|C_i|$ is the number of literals in clause C_i and l_{ik} is a literal, i.e., a propositional variable u_j or its negation $\overline{u_j}$, for $1 \le j \le n$. The set of clauses in the formula is denoted by $\mathbf{C}$. If one associates a weight w_i to each clause C_i one obtains the weighted *MAX–SAT* problem, denoted as *MAX W–SAT*: one is to determine the assignment of truth values to the n variables that maximizes the sum of the weights of the satisfied clauses. In the literature one often considers problems with different numbers k of literals per clause, defined as *MAX–k–SAT* , or *MAX W–k–SAT* in the weighted case. In some papers *MAX–k–SAT* instances contain up to k literals per clause, while in other papers they contain exactly k literals per clause. We consider the second option unless otherwise stated.

MAX–SAT is of considerable interest not only from the theoretical side but also from the practical one. On one hand, the decision version *SAT* was the first example of an *NP*–complete problem [16], moreover *MAX–SAT* and related variants play an important role in the characterization of different approximation classes like *APX* and *PTAS* [5]. On the other hand, many issues in mathematical logic and artificial intelligence can be expressed in the form of satisfiability or some of its variants, like constraint satisfaction. Some exemplary problems are consistency in expert system knowledge bases [46], integrity constraints in databases [4, 23], approaches to inductive inference [35, 40], asynchronous circuit synthesis [32]. An extensive review of algorithms for *MAX–SAT* appeared in [9].

`Davis and Putnam` [19] started in 1960 the investigation of useful strategies for handling resolution in the **Satisfiability** problem. `Davis, Logemann and Loveland` [18] avoid the memory explosion of the original *DP* algorithm by replacing the resolution rule with the *splitting rule*. A recent review of advanced techniques for resolution and splitting is presented in [31].

The *MAX W–SAT* problem has a natural **integer linear programming** formulation (*ILP*). Let $y_j = 1$ if Boolean variable u_j is **true**, $y_j = 0$ if it is **false**, and let the Boolean variable $z_i = 1$ if clause C_i is satisfied, $z_i = 0$ otherwise. The integer linear program is:

$$\max \sum_{i=1}^{m} w_i \, z_i$$

subject to the following constraints:

$$\sum_{j\in U_i^+} y_j + \sum_{j\in U_i^-} (1-y_j) \quad \ge \quad z_i \ , \ i = 1,\cdots,m$$

$$y_j \in \{0,1\} \ , \ j = 1,\cdots,n$$

$$z_i \in \{0,1\} \ , \ i = 1,\cdots,m$$

where U_i^+ and U_i^- denote the set of indices of variables that appear unnegated and negated in clause C_i, respectively. If one neglects the objective function and sets all z_i variables to 1, one obtains an integer programming feasibility problem associated to the *SAT* problem [11].

The integer linear programming formulation of *MAX–SAT* suggests that this problem could be solved by a *branch-and-bound* method. A usable method uses Chvátal cuts. In [35] it is

`Davis and Putnam`
Satisfiability
`Davis, Logemann and Loveland`
splitting rule
integer linear programming
branch-and-bound

shown that the resolvents in the propositional calculus correspond to certain *cutting planes* in the integer programming model of inference problems.

LP relaxations of integer linear programming formulations of *MAX–SAT* have been used to obtained upper bounds in [33, 55, 27]. A linear programming and rounding approach for *MAX 2–SAT* is presented in [13]. A method for strengthening the Generalized Set Covering formulation is presented in [47], where Lagrangian multipliers guide the generation of cutting planes.

The first **approximation algorithms** with a "guaranteed" quality of approximation [5] were proposed by `D. S. Johnson` [38] and use **greedy construction strategies**. The original paper [38] demonstrated for both of them a performance ratio 1/2. In detail, let k be the minimum number of variables occurring in any clause of the formula, $m(x,y)$ the number of clauses satisfied by the feasible solution y on instance x, and $m^*(x)$ the maximum number of clauses that can be satisfied.

For any integer $k \geq 1$, the first algorithm achieves a feasible solution y of an instance x such that

$$m(x,y)/m^*(x) \geq 1 - 1/k + 1.$$

while the second algorithm obtains

$$m(x,y)/m^*(x) \geq 1 - 1/2^k.$$

Recently it has been proved [12] that the second algorithm reaches a performance ratio 2/3. There are formulae for which the second algorithm finds a truth assignment such that the ratio is 2/3. Therefore this bound cannot be improved [12].

One of the most interesting approaches in the design of new algorithms is the use of *randomization*. During the computation, random bits are generated and used to influence the algorithm process. In many cases randomization allows to obtain better (expected) performance or to to simplify the construction of the algorithm. Two **randomized algorithms** that achieve a performance ratio of 3/4. have been proposed by `Goemans and Williamson` [27] and by `Yannakakis` [55]. Moreover, it is possible to derandomize these algorithms, that is, to obtain deterministic algorithms that preserve the same bound 3/4 for every instance. The approximation ratio 3/4 can be slightly improved [28]. Asano [2] (following [3]) has improved the bound to .77. For the restricted case of *MAX 2–SAT* , one can obtain a more substantial improvement (performance ratio 0.931) with the technique of `Feige and Goemans` [21]. If one considers only satisfiable *MAX W–SAT* instances, `Trevisan` [54] obtains a 0.8 approximation factor, while `Karloff and Zwick` [41] claim a 0.875 performance ratio for satisfiable instances of *MAX W–3–SAT*. A strong negative results about the approximability is due to `Håstad` [36]: Unless $P = NP$ *MAX W–SAT* cannot be approximated in polynomial time within a performance ratio greater than 7/8.

MAX–SAT is among the problems for which **local search** has been very successful: in practice, local search and its variations are the

cutting planes
LP relaxations
approximation algorithms
`D. S. Johnson`
greedy construction strategies
randomization
randomized algorithms
`Goemans and Williamson`
`Yannakakis`
`Feige and Goemans`
`Trevisan`
`Karloff and Zwick`
`Håstad`
local search

only efficient and effective method to address large and complex real-world instances. Different variations of local search with randomness techniques have been proposed for *SAT* and *MAX–SAT* starting from the late eighties, see for example [30, 52], motivated by previous applications of "min-conflicts" heuristics in the area of Artificial Intelligence [44].

The general scheme is based on generating a starting point in the set of admissible solution and trying to improve it through the application of basic moves. The search space is given by all possible truth assignments. Let us consider the elementary changes to the current assignment obtained by changing a single truth value. The definitions are as follows.

Let U be the discrete search space: $U = \{0,1\}^n$, and let f be the number of satisfied clauses. In addition, let $U^{(t)} \in U$ be the current configuration along the *search trajectory* at iteration t, and $N(U^{(t)})$ the neighborhood of point $U^{(t)}$, obtained by applying a set of basic moves μ_i $(1 \leq i \leq n)$, where μ_i complements the i-th bit u_i of the string: $\mu_i\ (u_1, u_2, ..., u_i, ..., u_n) = (u_1, u_2, ..., 1-u_i, ..., u_n)$.

$$N(U^{(t)}) = \{U \in U \text{ s.t. } U = \mu_i\, U^{(t)}, i = 1, ..., n\}$$

The version of **local search** (LS) that we consider starts from a random initial configuration $U^{(0)} \in U$ and generates a search trajectory as follows:

$$V = \textit{Best-Neighbor}(\ N(U^{(t)})\ ) \qquad (1)$$

$$U^{(t+1)} = \begin{cases} V & \textbf{if} \quad f(V) > f(U^{(t)}) \\ U^{(t)} & \textbf{if} \quad f(V) \leq f(U^{(t)}) \end{cases} \qquad (2)$$

where *Best-Neighbor* selects $V \in N(U^{(t)})$ with the best f value and ties are broken randomly. V in turn becomes the new current configuration if f improves. Other versions are satisfied with an improving (or non-worsening) neighbor, not necessarily the best one. Clearly, local search stops as soon as the first *local optimum* point is encountered, when no improving moves are available, see eqn. 2. Let us define as LS^+ a modification of LS where a specified number of iterations are executed and the candidate move obtained by *Best-Neighbor* is always accepted even if the f value remains equal or worsens.

Properties about the number of clauses satisfied at a *local optimum* have been demonstrated. Let m^* be the best value and k the minimum number of literals contained in the problem clauses. Let m_{loc} be the number of satisfied clauses at a local optimum of any instance of *MAX–SAT* with at least k literals per clause. m_{loc} satisfies the following bound [34]:

$$m_{loc} \geq (k/k+1)m$$

and the bound is sharp. Therefore, if m_{loc} is the number of satisfied clauses at a local optimum, then:

$$m_{loc} \geq (k/k+1)m^* \qquad (3)$$

State-of-the-art heuristics for *MAX–SAT* are obtained by complementing local search with schemes that are capable of producing better approximations beyond the locally optimal points. In some cases, these schemes generate a sequence of points in the set of admissible solutions in a way that is fixed before the search starts. An example is given by *multiple runs* of local search starting from different random points. The algorithm does not take into account the *history* of the previous phase of the search when the next points are generated. The term *memory-less* denotes this lack of feedback from the search history.

In addition to the cited multiple-run local search, these techniques are based on Markov processes (**Simulated Annealing**), "plateau"

search trajectory
local search
local optimum
local optimum
multiple runs
history
memory-less
Simulated Annealing

search and "random noise" strategies, or combinations of randomized constructions and local search. The use of a Markov process to generate a stochastic search trajectory is adopted for example in [53].

The *Gsat* algorithm was proposed in [52] as a *model-finding* procedure, i.e., to find an interpretation of the variables under which the formula comes out **true**. *Gsat* consists of multiple runs of LS^+, each run consisting of a number of iterations that is typically proportional to the problem dimension n. An empirical analysis of *Gsat* is presented in [25, 24]. Different "noise" strategies to escape from attraction basins are added to *Gsat* in [50, 51].

A hybrid algorithm that combines a randomized greedy construction phase to generate initial candidate solutions, followed be a local improvement phase is the *Grasp* scheme proposed in [48] for the *SAT* and generalized for the *MAX W–SAT* problem in [49]. *Grasp* is an iterative process, with each iteration consisting of two phases, a construction phase and a local search phase.

Different *history-sensitive heuristics* have been proposed to continue local search schemes beyond local optimality. These schemes aim at intensifying the search in promising regions and at diversifying the search into uncharted territories by using the information collected from the previous phase (the *history*) of the search. Because of the internal feedback mechanism, some algorithm parameters can be modified and tuned in an *on-line* manner, to reflect the characteristics of the *task* to be solved and the *local* properties of the configuration space in the neighborhood of the current point. This tuning has to be contrasted with the *off-line* tuning of an algorithm, where some parameters or choices are determined for a given problem in a preliminary phase and they remain fixed when the algorithm runs on a specific instance.

Tabu Search (*TS*) is a *history-sensitive* heuristic proposed by `F. Glover` [26] and, independently, by `Hansen and Jaumard`, that used the term *Samd* ("steepest ascent mildest descent") and applied it to the *MAX–SAT* problem in [34]. The main mechanism by which the history influences the search in *TS* is that, at a given iteration, some neighbors are *prohibited*, only a non-empty subset $N_A(U^{(t)}) \subset N(U^{(t)})$ of them is *allowed*. The general way of generating the search trajectory that we consider is given by:

$$N_A(U^{(t)}) = Allow(N(U^{(t)}), U^{(0)}, ..., U^{(t)}) \quad (4)$$

$$U^{(t+1)} = Best\text{-}Neighbor(\ N_A(U^{(t)})\ ) \quad (5)$$

The set-valued function *Allow* selects a non-empty subset of $N(U^{(t)})$ in a manner that depends on the entire previous history of the search $U^{(0)}, ..., U^{(t)}$. A specialized Tabu Search heuristic is used in [37] to speed up the search for a solution (if the problem is satisfiable) as part of a branch-and-bound algorithm for *SAT* , that adopts both a relaxation and a decomposition scheme by using polynomial instances, i.e., *2–SAT* and Horn- *SAT* .

Different methods to generate prohibitions produce *discrete dynamical systems* with qualitatively different *search trajectories*. In particular, prohibitions based on a list of *moves* lead to a faster escape from a locally optimal point than prohibitions based on a list of visited *configurations* [6]. In detail, the *Allow* function can be specified by introducing a *prohibition parameter* T (also called *list size*) that determines how

Gsat
model-finding
Grasp
history-sensitive heuristics
history
Tabu Search
history-sensitive
`F. Glover`
`Hansen and Jaumard`
discrete dynamical systems
search trajectories
prohibition parameter

long a move will remain prohibited after its execution. The *Fixed-TS* algorithm is obtained by fixing T throughout the search [26]. A neighbor is allowed if and only if it is obtained from the current point by applying a move that has not been used during the last T iterations. In detail, if $LU(\mu)$ is the last usage time of move μ ($LU(\mu) = -\infty$ at the beginning):

$$N_A(U^{(t)}) = \{U = \mu\, U^{(t)} \text{ s.t. } LU(\mu) < (t-T)\}$$

.

The *Reactive Tabu Search* algorithm of `Battiti and Tecchiolli` [10], *Reactive-TS* for short, defines simple rules to determine the prohibition parameter by reacting to the repetition of previously-visited configurations. One has a repetition if $U^{(t+R)} = U^{(t)}$, for $R \geq 1$. The prohibition period T depends on the iteration t and a *reaction* equation is added to the dynamical system:

$$T^{(t)} = \mathit{React}(T^{(t-1)}, U^{(0)}, ..., U^{(t)})$$

An algorithm that combines local search and *non-oblivious local search* [8], the use of prohibitions, and a reactive scheme to determine the prohibition parameter is the *Hamming-Reactive-TS* algorith proposed by `Battiti and Protasi` in [7], that contains also a detailed experimental analysis.

Given the hardness of the problem and the relevancy for applications in different fields, the emphasis on the **experimental analysis of algorithms** for the *MAX–SAT* problem has been growing in recent years.

In some cases the experimental comparisons have been executed in the framework of "challenges," with support of electronic collection and distribution of software, problem generators and test instances. An example is the the Second DIMACS Algorithm Implementation Challenge on Cliques, Coloring and Satisfiability, whose results have been published in [39]. Practical and industrial *MAX–SAT* problems and benchmarks, with significant case studies are also presented in [20]. Some basic problem models that are considered both in theoretical and in experimental studies of *MAX–SAT* algorithms are described in [31].

Different algorithms demonstrate a different degree of effort, measured by number of elementary steps or CPU time, when solving different kinds of instances. For example, `Mitchell et al.` [45] found that some distributions used in past experiments are of little interest because the generated formulae are almost always very easy to satisfy. They also reported that one can generate very hard instances of k-SAT, for $k \geq 3$. In addition, they report the following observed behavior for random fixed length *3–SAT* formulae: if r is the ratio r of clauses to variables ($r = m/n$), almost all formulae are satisfiable if $r < 4$, almost all formulae are unsatisfiable if $r > 4.5$. A rapid transition seems to appear for $r \approx 4.2$, the same point where the computational complexity for solving the generated instances is maximized, see [42, 17] for reviews of experimental results.

Let κ be the least real number such that, if r is larger than κ, then the probability of $\mathbf{C}$ being satisfiable converges to 0 as n tends to infinity. A notable result found independently by many people, including [22] and [15] is that

$$\kappa \leq \log_{8/7} 2 = 5.191$$

A series of theoretical analyses aim at approximating the *unsatisfiability threshold* of random formulae [43, 1, 14, 29].

References

[1] Achlioptas, D., Kirousis, L. M., Kranakis, E., and Krinzac, D.: 'Rigorous results for random (2+p)-SAT': *Proc. Work. on Randomized Algorithms in Sequential, Parallel and Distributed Computing (RALCOM 97), Santorini, Greece*, 1997, pp. 1–10.

Reactive Tabu Search
`Battiti and Tecchiolli`
non-oblivious local search
Hamming-Reactive-TS
`Battiti and Protasi`
experimental analysis of algorithms
`Mitchell et al.`
unsatisfiability threshold

[2] Asano, T.: 'Approximation algorithms for MAX-SAT: Yannakakis vs. Goemans-Williamson': *Proc. 3rd Israel Symp. on the Theory of Computing and Systems, Ramat Gan, Israel*, 1997, pp. 24–37.

[3] Asano, T., Ono, T., and Hirata, T.: 'Approximation algorithms for the maximum satisfiability problem': *Proc. 5th Scandinavian Work. on Algorithms Theory*, 1996, pp. 110–111.

[4] Asirelli, P., Santis, M. de, and Martelli, A.: 'Integrity constraints in logic databases', *Journal of Logic Programming* **3** (1985), 221–232.

[5] Ausiello, G., Crescenzi, P., and Protasi, M.: 'Approximate solution of NP optimization problems', *Theoretical Computer Science* **150** (1995), 1–55.

[6] Battiti, R.: 'Reactive Search: Toward Self-Tuning Heuristics', *Modern Heuristic Search Methods*, in V. J. Rayward-Smith, I. H. Osman, C. R. Reeves, and G. D. Smith (eds.). John Wiley and Sons, 1996, ch. 4, pp. 61–83.

[7] Battiti, R., and Protasi, M.: 'Reactive Search, a history-sensitive heuristic for MAX-SAT', *ACM Journal of Experimental Algorithmics* **2**, no. 2 (1997).

[8] Battiti, R., and Protasi, M.: 'Solving MAX-SAT with non-oblivious functions and history-based heuristics', *Satisfiability Problem: Theory and Applications*, no. 35 in DIMACS: Series in Discrete Mathematics and Theoretical Computer Science. AMS and ACM Press, 1997.

[9] Battiti, R., and Protasi, M.: 'Approximate Algorithms and Hauristics for MAX-SAT', *Handbook of Combinatorial Optimization*, in D.-Z. Du and P.M. Pardalos (eds.). Kluwer Academic Publishers, 1998, pp. 77–148.

[10] Battiti, R., and Tecchiolli, G.: 'The Reactive Tabu Search', *ORSA Journal on Computing* **6**, no. 2 (1994), 126–140.

[11] Blair, C.E., Jeroslow, R.G., and Lowe, J.K.: 'Some results and experiments in programming for propositional logic', *Computers and Operations Research* **13**, no. 5 (1986), 633–645.

[12] Chen, J., Friesen, D., and Zheng, H.: 'Tight bound on Johnson's algorithm for MAX-SAT': *Proc. 12th Annual IEEE Conf. on Computational Complexity, Ulm, Germany*, 1997, pp. 274–281.

[13] Cheriyan, J., Cunningham, W. H., Tuncel, T., and Wang, Y.: 'A linear programming and rounding approach to MAX 2-SAT', in M. Trick and D. S. Johson (eds.): *Proc. of the Second DIMACS Algorithm Implementation Challenge on Cliques, Coloring and Satisfiability*, no. 26 in DIMACS Series on Discrete Mathematics and Theoretical Computer Science, 1996, pp. 395–414.

[14] Chvatal, V., and Reed, B.: 'Mick gets some (the odds are on his side)': *Proc. 33th Ann. IEEE Symp. on Foundations of Computer Science*, IEEE Computer Society, 1992, pp. 620–627.

[15] Chvátal, V., and Szemerédi, E.: 'Many hard examples for resolution', *Journal of the ACM* **35** (1988), 759–768.

[16] Cook, S.A.: 'The complexity of theorem-proving procedures': *Proc. of the Third Annual ACM Symp. on the Theory of Computing*, 1971, pp. 151–158.

[17] Cook, S.A., and Mitchell, D.G.: 'Finding hard instances of the satisfiability problem: a survey', *Satisfiability Problem: Theory and Applications*, in D.-Z. Du, J. Gu, and P.M. Pardalos (eds.), Vol. 35 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. AMS and ACM Press, 1997, pp. 1–17.

[18] Davis, M., Logemann, G., and Loveland, D.: 'A machine program for theorem proving', *Communications of the ACM* **5** (1962), 394–397.

[19] Davis, M., and Putnam, H.: 'A computing procedure for quantification theory', *Journal of the ACM* **7** (1960), 201–215.

[20] Du, D., Gu, J., and (Eds.), P.M. Pardalos: *Satisfiability Problem: Theory and Applications*, Vol. 35 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, AMS and ACM Press, 1997.

[21] Feige, U., and Goemans, M.X.: 'Approximating the value of two proper proof systems, with applications to MAX-2SAT and MAX-DICUT': *Proc. of the Third Israel Symp. on Theory of Computing and Systems*, 1995, pp. 182–189.

[22] Franco, J., and Paull, M.: 'Probabilistic analysis of the Davis-Putnam procedure for solving the satisfiability problem', *Discrete Applied Mathematics* **5** (1983), 77–87.

[23] Gallaire, H., Minker, J., and Nicolas, J. M.: 'Logic and databases: a deductive approach', *Computing Surveys* **16**, no. 2 (1984), 153–185.

[24] Gent, I.P., and Walsh, T.: 'An empirical analysis of search in GSAT', *Journal of Artificial Intelligence Research* **1** (1993), 47–59.

[25] Gent, I.P., and Walsh, T.: 'Towards an understanding of hill-climbing procedures for SAT': *Proc. of the Eleventh National Conf. on Artificial Intelligence*, AAAI Press / The MIT Press, 1993, pp. 28–33.

[26] Glover, F.: 'Tabu Search - part I', *ORSA Journal on Computing* **1**, no. 3 (1989), 190–260.

[27] Goemans, M.X., and Williamson, D.P.: 'New 3/4-approximation algorithms for the maximum satisfiability problem', *SIAM Journal on Discrete Mathematics* **7**, no. 4 (1994), 656–666.

[28] Goemans, M.X., and Williamson, D.P.: 'Improved approximation algorithms for maximum cut

and satisfiability problems using semidefinite programming', *Journal of the ACM* **42**, no. 6 (1995), 1115–1145.

[29] GOERDT, A.: 'A threshold for unsatisfiability', *Journal of Computer and System Sciences* **53** (1996), 469–486.

[30] GU, J.: 'Efficient local search for very large-scale satisfiability problem', *ACM SIGART Bulletin* **3**, no. 1 (1992), 8–12.

[31] GU, J., PURDOM, P.W., FRANCO, J., AND WAH, B.W.: 'Algorithms for the Satisfiability (SAT) Problem: A survey', *Satisfiability Problem: Theory and Applications*, in D.-Z. DU, J. GU, AND P.M. PARDALOS (eds.), Vol. 35 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science.* AMS and ACM Press, 1997.

[32] GU, J., AND PURI, R.: 'Asynchronous Circuit Synthesis with Boolean Satisfiability', *IEEE Transactions on Computer-Aided Design of Integrated Circuits* **14**, no. 8 (1995), 961–973.

[33] HAMMER, P.L., HANSEN, P., AND SIMEONE, B.: 'Roof duality, complementation and persistency in quadratic 0-1 optimization', *Mathematical Programming* **28** (1984), 121–155.

[34] HANSEN, P., AND JAUMARD, B.: 'Algorithms for the maximum satisfiability problem', *Computing* **44** (1990), 279–303.

[35] HOOKER, J.N.: 'Resolution vs. cutting plane solution of inference problems: some computational experience', *Operations Research Letters* **7**, no. 1 (1988), 1–7.

[36] HÅSTAD, J.: 'Some optimal inapproximability results': *Proc. 28th Annual ACM Symp. on Theory of Computing, El Paso, Texas*, 1997, pp. 1–10.

[37] JAUMARD, B., STAN, M., AND DESROSIERS, J.: 'Tabu search and a quadratic relaxation for the satisfiability problem', in M. TRICK AND D. S. JOHSON (eds.): *Proc. of the Second DIMACS Algorithm Implementation Challenge on Cliques, Coloring and Satisfiability*, no. 26 in DIMACS Series on Discrete Mathematics and Theoretical Computer Science, 1996, pp. 457–477.

[38] JOHNSON, D.S.: 'Approximation algorithms for combinatorial problems', *Journal of Computer and System Sciences* **9** (1974), 256–278.

[39] JOHNSON, D.S., AND (EDS.), M. TRICK: *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge*, Vol. 26 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, AMS, 1996.

[40] KAMATH, A.P., KARMARKAR, N.K., RAMAKRISHNAN, K.G., AND RESENDE, M.G.: 'Computational exprience with an interior point algorithm on the satisfiability problem', *Annals of Operations Research* **25** (1990), 43–58.

[41] KARLOFF, H., AND ZWICK, U.: 'A 7/8-Approximation Algorithm for MAX 3SAT?': *Proc. of the 38th Annual IEEE Symp. on Foundations of Computer Science*, IEEE Computer Society, 1997.

[42] KIRKPATRICK, S., AND SELMAN, B.: 'Critical Behavior in the Satisfiability of Random Boolean Expressions', *Science* **264** (1994), 1297–1301.

[43] KIROUSIS, L. M., KRANAKIS, E., AND KRIZANC, D.: 'Approximating the Unsatisfiability Threshold of Random Formulas': *Proc. of the Fourth Annual European Symp. on Algorithms*, Springer-Verlag, September 1996, pp. 27–38.

[44] MINTON, S., JOHNSTON, M. D., PHILIPS, A. B., AND LAIRD, P.: 'Solving large-scale constraint satisfaction and scheduling problems using a heuristic repair method': *Proc. of the 8th National Conf. on Artificial Intelligence (AAAI-90)*, 1990, pp. 17–24.

[45] MITCHELL, D., SELMAN, B., AND LEVESQUE, H.: 'Hard and easy distributions of SAT problems': *Proc. of the 10th National Conf. on Artificial Intelligence (AAAI-92)*, July 1992, pp. 459–465.

[46] NGUYEN, T.A., PERKINS, W.A., LAFFREY, T.J., AND PECORA, D.: 'Checking an expert system knowledge base for consistency and completeness': *Proc. of the International Joint Conf. on Artificial Intelligence*, 1985, pp. 375–378.

[47] NOBILI, P., AND SASSANO, A.: Strengthening Lagrangian Bounds for the MAX-SAT problem, Tech. Rep. 96-230, Institut fuer Informatik, Koln Univ., Germany, 1996, Proc. of the Work. on the Satisfiability Problem, Siena, Italy (J. Franco and G. Gallo and H. Kleine Buening, Eds.).

[48] RESENDE, M.G.C., AND FEO, T. A.: 'A GRASP for Satisfiability', in M. TRICK AND D. S. JOHSON (eds.): *Proc. of the Second DIMACS Algorithm Implementation Challenge on Cliques, Coloring and Satisfiability*, no. 26 in DIMACS Series on Discrete Mathematics and Theoretical Computer Science, 1996, pp. 499–520.

[49] RESENDE, M.G.C., PITSOULIS, L.S., AND PARDALOS, P.M.: 'Approximate solution of weighted MAX-SAT problems using GRASP', *Satisfiability Problem: Theory and Applications*, no. 35 in DIMACS: Series in Discrete Mathematics and Theoretical Computer Sc ience. American Mathematical Society, 1997.

[50] SELMAN, B., AND KAUTZ, H.: 'Domain-Independent Extensions to GSAT: Solving Large Structured Satisfiability Problems': *Proc. of the International Joint Conf. on Artificial Intelligence*, 1993, pp. 290–295.

[51] SELMAN, B., KAUTZ, H.A., AND COHEN, B.: 'Local search strategies for satisfiability testing', in M. TRICK AND D. S. JOHSON (eds.): *Proc. of the Second DIMACS Algorithm Implementation Challenge on Cliques, Coloring and Satisfiability*, no. 26

in DIMACS Series on Discrete Mathematics and Theoretical Computer Science, 1996, pp. 521–531.

[52] SELMAN, B., LEVESQUE, H., AND MITCHELL, D.: 'A new method for solving hard satisfiability problems': *Proc. of the 10th National Conf. on Artificial Intelligence (AAAI-92)*, July 1992, pp. 440–446.

[53] SPEARS, W.M.: 'Simulated Annealing for Hard Satisfiability Problems', in M. TRICK AND D. S. JOHNSON (eds.): *Proc. of the Second DIMACS Algorithm Implementation Challenge on Cliques, Coloring and Satisfiability*, no. 26 in DIMACS Series on Discrete Mathematics and Theoretical Computer Science, 1996, pp. 533–555.

[54] TREVISAN, L.: 'Approximating Satisfiable Satisfiability Problems': *Proc. of the 5th Annual European Symp. on Algorithms, Graz*, Springer Verlag, 1997, pp. 472–485.

[55] YANNAKAKIS, M.: 'On the approximation of maximum satisfiability', *Journal of Algorithms* **17** (1994), 475–502.

Roberto Battiti
Dip. di Matematica, Univ. di Trento
Via Sommarive, 14
38050 Povo (Trento)
Italy
E-mail address: battiti@science.unitn.it

AMS 1991 Subject Classification: Primary 03B05, 68Q25, 90C09; Secondary 90C27, 68P10, 68R05, 68T15, 68T20, 94C10.

Key words and phrases: Maximum Satisfiability, local search, approximation algorithms, history-sensitive heuristics.