Handbook of Satisfiability
Editors A. Biere, M. Heule, H. van Maaren, and T. Walsh
IOS Press, 2008

Chapter 6

Incomplete Algorithms [draft v2]

Henry Kautz, Ashish Sabharwal, and Bart Selman

An *incomplete* method for solving the propositional satisfiability problem (or a general constraint satisfaction problem) is one that does not provide the guarantee that it will eventually either report a satisfying assignment or declare that the given formula is unsatisfiable. In practice, most such methods are biased towards the satisfiable side: they are typically run with a pre-set limit, after which they either produce a solution or report failure; they never declare the formula to be unsatisfiable. These are the kind of algorithms we will discuss in this chapter. In principle, an incomplete algorithm could instead be biased towards the unsatisfiable side, always providing proofs of unsatisfiability but failing to find solutions to some satisfiable instances, or be incomplete w.r.t. both satisfiable and unsatisfiable instances.

Unlike systematic solvers often based on an exhaustive branching and backtracking search, incomplete methods are generally based on *stochastic local search*, sometimes referred to as SLS. On problems from a variety of domains, such incomplete methods for SAT can significantly outperform DPLL-based methods. Since the early 1990's, there has been a tremendous amount of research on designing, understanding, and improving local search methods for SAT.[1] There have also been attempts at hybrid approaches that explore combining ideas from DPLL methods and local search techniques [e.g. Habet et al., 2002, Mazure et al., 1996, Prestwich, 2001, Rish and Dechter, 1996]. We cannot do justice to all recent research in local search solvers for SAT, and will instead try to provide a brief overview and touch upon some interesting details. The interested reader is encouraged to further explore the area through some of the nearly a hundred publications we cite along the way.

We begin the chapter by discussing two methods that played a key role in the success of local search for satisfiability, namely `GSAT` [Selman et al., 1992]

[1] For example, there is work by Anbulagan et al. [2005], Cha and Iwama [1996], Frank et al. [1997], Gent and Walsh [1993], Ginsberg and McAllester [1994], Gu [1992], Gu et al. [1997], Heule and van Maaren [2006], Heule et al. [2004], Hirsch and Kojevnikov [2005], Hoos [1999, 2002], Hoos and Stützle [2004], Kirkpatrick and Selman [1994], Konolige [1994], Li et al. [2007, 2003], McAllester et al. [1997], Morris [1993], Parkes and Walser [1996], Pham et al. [2007], Resende and Feo [1996], Schuurmans and Southey [2001], Spears [1996], Thornton et al. [2004], Wu and Wah [2000], and others.

and `Walksat` [Selman, Kautz, and Cohen, 1996]. We will then explore alternative techniques based on the discrete Lagrangian method [Schuurmans et al., 2001, Shang and Wah, 1998, Wu and Wah, 1999, 2000], the phase transition phenomenon in random k-SAT [Crawford and Auton, 1993, Kirkpatrick and Selman, 1994, Mitchell et al., 1992], and a relatively new incomplete technique called Survey Propagation [Mézard et al., 2002].

We note that there are other exciting related solution methodologies, such as those based on translation to integer programming [Hooker, 1988, Kamath et al., 1990], that we will not discuss here. There has also been work on formally analyzing local search methods, yielding some of the best $o(2^n)$ time algorithms for SAT. For instance, the expected running time, ignoring polynomial factors, of a simple local search method with restarts after every $3n$ "flips" has been shown to be $(2 \cdot (k-1)/k)^n$ for k-SAT [Schöning, 1999, 2002], which yields a complexity of $(4/3)^n$ for 3-SAT. This result has been derandomized to yield a deterministic algorithm with complexity 1.481^n up to polynomial factors [Dantsin et al., 2002].

For the discussion in the rest of this chapter, it will be illustrative to think of a propositional formula F with n variables and m clauses as creating a discrete manifold or *landscape* in the space $\{0,1\}^n \times \{0,1,\ldots,m\}$. The 2^n truth assignments to the variables of F correspond to the points in $\{0,1\}^n$, and the "height" of each such point, a number in $\{0,1,\ldots,m\}$, corresponds to the number of unsatisfied clauses of F under this truth assignment. The solutions or satisfying assignments for F are precisely the points in this landscape with height zero, and thus correspond to the global minima of this landscape, or, equivalently, the global minima of the function that assigns each point in $\{0,1\}^n$ to its height. The search problem for SAT is then a search for a global minimum in this implicitly defined exponential-size landscape. Clearly, if the landscape did not have any local minima, a greedy descent would provide an effective search method. All interesting problems, however, do have local minima—the main challenge and opportunity for the designers of local search methods.

This landscape view also leads one naturally to the problem of *maximum satisfiability* or MAX-SAT: Given a formula F, find a truth assignment that satisfies the most number of clauses possible. Solutions to the MAX-SAT problem are, again, precisely the global minima of the corresponding landscape, only that these global minima may not have height zero. Most of the incomplete methods we will discuss in this chapter, especially those based on local search, can also work as a solution approach for the MAX-SAT problem, by providing the "best found" truth assignment (i.e., one with the lowest observed height) upon termination. Of course, while it is easy to test whether the best found truth assignment is a solution to a SAT instance, it is NP-hard to perform the same test for a MAX-SAT instance. Thus, this approach only provides a heuristic algorithm for MAX-SAT. For further details on this problem, we refer the reader to Part 2, Chapter 5 of this Handbook.

6.1. Greedy Search, Focused Random Walk, and Extensions

The original impetus for trying a local search method on satisfiability problems was the successful application of such methods for finding solutions to large N-

queens problems, first using a connectionist system by Adorf and Johnston [1990], and then using greedy local search by Minton et al. [1990]. It was originally assumed that this success simply indicated that N-queens was an easy problem, and researchers felt that such techniques would fail in practice for SAT and other more intricate problems. In particular, it was believed that local search methods would easily get stuck in local minima, with a few clauses remaining unsatisfied. Experiments with the solver `GSAT` showed, however, that certain local search strategies often do reach global minima, in many cases much faster than systematic search methods.

`GSAT` is based on a randomized local search technique [Lin and Kernighan, 1973, Papadimitriou and Steiglitz, 1982]. The basic `GSAT` procedure, introduced by Selman et al. [1992] and described here as Algorithm 6.1, starts with a randomly generated truth assignment for all variables. It then greedily changes ('flips') the truth assignment of the variable that leads to the greatest decrease in the total number of unsatisfied clauses. The *neighborhood* of the current truth assignment, thus, is the set of n truth assignments that differ from the current one in exactly one variable. Such flips are repeated until either a satisfying assignment is found or a pre-set maximum number of flips (MAX-FLIPS) is reached. This process is repeated as needed, up to a maximum of MAX-TRIES times.

Algorithm 6.1: `GSAT` (F)

```
Input      : A CNF formula F
Parameters : Integers MAX-FLIPS, MAX-TRIES
Output     : A satisfying assignment for F, or FAIL
begin
  for i ← 1 to MAX-TRIES do
    σ ← a randomly generated truth assignment for F
    for j ← 1 to MAX-FLIPS do
      if σ satisfies F then return σ                    // success
      v ← a variable flipping which results in the greatest decrease
            (possibly negative) in the number of unsatisfied clauses
      Flip v in σ
  return FAIL                       // no satisfying assignment found
end
```

Selman et al. showed that `GSAT` substantially outperformed even the best backtracking search procedures of the time on various classes of formulas, including randomly generated formulas and SAT encodings of graph coloring problems [Johnson et al., 1991]. The search of `GSAT` typically begins with a rapid greedy descent towards a better truth assignment (i.e., one with a lower height), followed by long sequences of "sideways" moves. Sideways moves are moves that do not increase or decrease the total number of unsatisfied clauses. In the landscape corresponding to the formula, each collection of truth assignments that are connected together by a sequence of possible sideways moves is referred to as a *plateau*. Experiments indicate that on many formulas, `GSAT` spends most of its time on plateaus, transitioning from one plateau to another every so often. Interestingly, Frank et al. [1997] observed that in practice, almost all plateaus do have so-called "exits" that lead to another plateau with a lower number of unsatisfied clauses. Intuitively, in a very high dimensional search space such as the space of a 10,000 variable formula, it is very rare to encounter local minima, which are plateaus from where there is no local move that decreases the number of unsatisfied clauses. In practice, this means that `GSAT` most often does not get stuck in local minima, although it may take a substantial amount of time on each plateau before moving on to the next one. This motivates studying various modifications in order to speed up this process [Selman and Kautz, 1993, Selman et al., 1994]. One of the most successful strategies is to introduce noise into the search in the form of uphill moves, which forms the basis of the now well-known local search method for SAT called `Walksat` [Selman et al., 1996].

`Walksat` interleaves the greedy moves of `GSAT` with random walk moves of a standard Metropolis search. It further *focuses the search* by always selecting the variable to flip from an unsatisfied clause C (chosen at random). This seemingly simple idea of focusing the search turns out to be crucial for scaling such techniques to problems beyond a few hundred variables. If there is a variable in C flipping which does not turn any currently satisfied clauses to unsatisfied, it flips this variable (a "freebie" move). Otherwise, with a certain probability, it flips a random literal of C (a "random walk" move), and with the remaining probability, it flips a variable in C that minimizes the *break-count*, i.e., the number of currently satisfied clauses that become unsatisfied (a "greedy" move). `Walksat` is presented in detail as Algorithm 6.2. One of its parameters, in addition to the maximum number of tries and flips, is the *noise* $p \in [0, 1]$, which controls how often are non-greedy moves considered during the stochastic search. It has been found empirically that for various problems from a single domain, a single value of p is optimal. For random 3-SAT formulas, the optimal noise is seen to be 0.57, and at this setting, `Walksat` is empirically observed to scale linearly even for clause-to-variable ratios $\alpha > 4.2$ [Seitz et al., 2005].[2]

The focusing strategy of `Walksat` based on selecting variables solely from unsatisfied clauses was inspired by the $\mathrm{O}(n^2)$ randomized algorithm for 2-SAT by Papadimitriou [1991]. It can be shown that for any satisfiable formula and starting from any truth assignment, there exists a sequence of flips using only variables from unsatisfied clauses such that one obtains a satisfying assignment.

When one compares the biased random walk strategy of `Walksat` on hard random 3-CNF formulas against basic `GSAT`, the simulated annealing process of Kirkpatrick et al. [1983], and a pure random walk strategy, the biased random walk process significantly outperforms the other methods [Selman et al., 1994]. In the years following the development of `Walksat`, many similar methods have been shown to be highly effective on not only random formulas but on several classes of structured instances, such as encodings of circuit design problems, Steiner tree problems, problems in finite algebra, and AI planning [cf. Hoos and Stützle, 2004]. Various extensions of the basic process have also been explored, such as dynamic search policies like `adapt-novelty` [Hoos, 2002], incorporating unit clause elimination as in the solver `UnitWalk` [Hirsch and Kojevnikov, 2005], and exploiting

[2] Aurell et al. [2004] had observed earlier that `Walksat` scales linearly for random 3-SAT at least till clause-to-variable ratio 4.15.

Algorithm 6.2: `Walksat` (F)

```
Input      : A CNF formula F
Parameters : Integers MAX-FLIPS, MAX-TRIES; noise parameter p ∈ [0, 1]
Output     : A satisfying assignment for F, or FAIL
begin
    for i ← 1 to MAX-TRIES do
        σ ← a randomly generated truth assignment for F
        for j ← 1 to MAX-FLIPS do
            if σ satisfies F then return σ                      // success
            C ← an unsatisfied clause of F chosen at random
            if ∃ variable x ∈ C with break-count = 0 then
                v ← x                                           // freebie move
            else
                With probability p:                             // random walk move
                    v ← a variable in C chosen at random
                With probability 1 − p:                         // greedy move
                    v ← a variable in C with the smallest break-count
            Flip v in σ
    return FAIL                              // no satisfying assignment found
end
```

---

problem structure for increased efficiency [Pham et al., 2007]. Recently, it was shown that the performance of stochastic solvers on many structured problems can be further enhanced by using new SAT encodings that are designed to be effective for local search [Prestwich, 2007].

While adding random walk moves as discussed above turned out to be a successful method of guiding the search away from local basins of attraction and toward other parts of the search space, a different line of research considered techniques that relied on the idea of *clause re-weighting* as an extension of basic greedy search [Cha and Iwama, 1996, Frank, 1997, Hutter et al., 2002, Morris, 1993, Selman et al., 1992, Thornton et al., 2004]. Here one assigns a positive weight to each clause and attempts to minimize the sum of the weights of the unsatisfied clauses. The clause weights are dynamically modified as the search progresses, increasing the weight of the clauses that are currently unsatisfied. (In some implementations, increasing the weight of a clause was done by simply adding identical copies of the clause.) In this way, if one waits sufficiently long, any unsatisfied clause gathers enough weight so as to sway the truth assignment in its favor. This is thought of as "flooding" the current local minimum by re-weighting unsatisfied clauses to create a new descent direction for local search. Variants of this approach differ in the re-weighting strategy used, e.g., how often and by how much the weights of unsatisfied clauses are increased, and how are all weights periodically decreased in order to prevent certain weights from becoming dis-proportionately high. The work on DLM or Discrete Lagrangian Method grounded these techniques in a solid theoretical framework, whose details we defer to Section 6.2. The SDF or "smoothed descent and flood" system of Schuurmans and Southey [2001] achieved significantly improved results by using multiplicative (rather than additive) re-weighting, by making local moves based on how strongly clauses are currently satisfied in terms of the number of satisfied literals (rather than simply how many are satisfied), and by periodically shrinking all weights towards their common mean.

Other approaches for improving the performance of `GSAT` were also explored. These included `TSAT` by Mazure et al. [1997], who maintained a tabu list in order to prevent `GSAT` from repeating earlier moves, and `HSAT` by Gent and Walsh [1993], who considered breaking ties in favor of least recently flipped variables. These strategies provided improvement, but to a lesser extent than the random walk component added by `Walksat`.

In an attempt towards understanding many of these techniques better, Schuurmans and Southey [2001] proposed three simple, intuitive measures of the effectiveness of local search: *depth, mobility,* and *coverage*. (A) Depth measures how many clauses remain unsatisfied as the search proceeds. Typically, good local search strategies quickly descend to low depth and stay there for a long time; this corresponds to spending as much time as possible near the bottom of the search landscape. (B) Mobility measures how rapidly the process moves to new regions in the search space (while simultaneously trying to stay deep in the objective). Clearly, the larger the mobility, the better chance a local search strategy has of success. (C) Coverage measures how systematically the process explores the entire space, in terms of the largest "gap", i.e., the maximum Hamming distance between any unexplored assignment and the nearest evaluated assignment. Schuurmans and Southey [2001] hypothesized that, in general, successful local search procedures work well not because they possess any special ability to predict whether a local basin of attraction contains a solution or not—rather they simply descend to promising regions and explore near the bottom of the objective as rapidly, broadly, and systematically as possible, until they stumble across a solution.

6.2. Discrete Lagrangian Methods

Shang and Wah [1998] introduced a local search system for SAT based on the theory of Lagrange multipliers. They extended the standard Lagrange method, traditionally used for continuous optimization, to the discrete case of propositional satisfiability, in a system called DLM (Discrete Lagrangian Method). Although the final algorithm that comes out of this process can be viewed as a "clause weighted" version of local search as discussed earlier, this approach provided a theoretical foundation for design choices that had appeared somewhat ad-hoc in the past. The change in the weights of clauses that are unsatisfied translates in this system to a change is the corresponding Lagrange multipliers, as one searches for a (local) optimum of the associated Lagrange function.

The basic idea behind DLM is the following. Consider an n-variable CNF formula F with clauses $C_1, C_2, \ldots, C_m$. For $x \in \{0,1\}^n$ and $i \in \{1, 2, \ldots, m\}$, let $U_i(x)$ be a function that is 0 if C_i is satisfied by x, and 1 otherwise. Then the SAT problem for F can be written as the following optimization problem over

$x \in \{0,1\}^n$:

$$\begin{aligned} \text{minimize} \quad & N(x) = \sum_{i=1}^{m} U_i(x) & & (6.1) \\ \text{subject to} \quad & U_i(x) = 0 & \forall i \in \{1, 2, \ldots, m\} & \end{aligned}$$

Notice that $N(x) \geq 0$ and equals 0 iff all clauses of F are satisfied. Thus, the objective function $N(x)$ is minimized iff x is a satisfying assignment for F. This formulation, somewhat strangely, also has each clause as an explicit constraint $U_i(x)$, so that any feasible solution is automatically also locally as well as globally optimal. This redundancy, however, is argued to be the strength of the system: the dynamic shift in emphasis between the objective and the constraints, depending on the relative values of the Lagrange multipliers, is the key feature of Lagrangian methods.

The theory of discrete Lagrange multipliers provides a recipe for converting the above constrained optimization problem into an unconstrained optimization problem, by introducing a Lagrange multiplier for each of the constraints and adding the constraints, appropriately multiplied, to the objective function. The resulting discrete Lagrangian function, which provides the new objective function to be optimized, is similar to what one would obtain in the traditional continuous optimization case:

$$L_d(x, \lambda) = N(x) + \sum_{i=1}^{m} \lambda_i U_i(x) \tag{6.2}$$

where $x \in \{0,1\}^n$ are points in the variable space and $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_m) \in \mathbb{R}^m$ is a vector of Lagrange multipliers, one for each constraint in (i.e., clause of) F. A point $(x^*, \lambda^*) \in \{0,1\}^n \times \mathbb{R}^m$ is called a *saddle point* of the Lagrange function $L_d(x, \lambda)$ if it is a local minimum w.r.t. x^* and a local maximum w.r.t. λ^*. Formally, (x^*, λ^*) is a saddle point for L_d if

$$L_d(x^*, \lambda) \;\leq\; L_d(x^*, \lambda^*) \;\leq\; L_d(x, \lambda^*)$$

for all λ sufficient close to λ^* and for all x that differ from x^* only in one dimension. It can be proven that x^* is a local minimum solution to the discrete constrained optimization formulation of SAT (6.1) if there exists λ^* such that (x^*, λ^*) is a saddle point of the associated Lagrangian function $L_d(x, \lambda)$. Therefore, local search methods based on the Lagrangian system look for saddle points in the extended space of variables and Lagrange multipliers. By doing descents in the original variable space and ascents in the Lagrange-multiplier space, a saddle point equilibrium is reached when (locally) optimal solutions are found.

For SAT, this is done through a *difference gradient* $\Delta_x L_d(x, \lambda)$, defined to be a vector in $\{-1, 0, 1\}^n$ with the properties that it has at most one non-zero entry and $y = x \oplus \Delta_x L_d(x, \lambda)$ minimizes $L_d(y, \lambda)$ over all neighboring points y of x, including x itself. Here $\oplus$ is the vector addition; $x \oplus z = (x_1 + z_1, \ldots, x_n + z_n)$. The neighbors are "user-defined" and are usually taken to be all points that differ from x only in one dimension (i.e., are one variable flip away). Intuitively, $\Delta_x L_d(x, \lambda)$ "points in the direction" of the neighboring value in the variable space that minimizes the Lagrangian function for the current λ.

This yields an algorithmic approach for minimizing the discrete Lagrangian function $L_d(x, \lambda)$ associated with SAT (and hence minimizing the objective function $N(x)$ in the discrete constrained optimization formulation of SAT (6.1)). The algorithm proceeds iteratively in stages, updating $x \in \{0,1\}^n$ and $\lambda \in \mathbb{R}^m$ in each stage using the difference gradient and the current status of each constraint in terms of whether or not it is satisfied, until a fixed point is found. Let $x(k)$ and $\lambda(k)$ denote the values of x and λ after the k^{th} iteration of the algorithm. Then the updates are as follows:

$$\begin{aligned} x(k+1) &= x(k) \oplus \Delta_x L_d(x(k), \lambda(k)) \\ \lambda(k+1) &= \lambda(k) + c\, U(x(k)) \end{aligned}$$

where $c \in \mathbb{R}^+$ is a parameter that controls how fast the Lagrange multipliers are increased over iterations and U denotes the vector of the m constraint functions U_i defined earlier. The difference gradient $\Delta_x L_d$ determines which variable, if any, to flip in order to lower $L_d(x, \lambda)$ for the current λ. When a fixed point for these iterations is reached, i.e., when $x(k+1) = x(k)$ and $\lambda(k+1) = \lambda(k)$, it must be that all constraints U_i are satisfied. To see this, observe that if the i^{th} clause is unsatisfied after the k^{th} iteration, then $U_i(x(k)) = 1$, which implies $\lambda_i(k+1) = \lambda_i(k) + c$; thus, λ_i will keep increasing until U_i is satisfied. This provides dynamic clause re-weighting in this system, placing more and more emphasis on clauses that are unsatisfied until they become satisfied. Note that changing the Lagrange multipliers λ_i in turn affects x by changing the direction in which the difference gradient $\Delta_x L_d(x, \lambda)$ points, eventually leading to a point at which all constraints are satisfied. This is the essence of the DLM system for SAT.

The basic implementation of DLM [Shang and Wah, 1998] uses a simple controlled update rule for λ: increase λ_i by 1 for all unsatisfied clauses C_i after a pre-set number θ_1 of *up-hill* or *flat* moves (i.e., changes in x that do not decrease $L_d(x, \lambda)$) have been performed. In order to avoid letting some Lagrange multipliers become disproportionately large during the search, all λ_i's are periodically decreased after a pre-set number θ_2 of increases in λ have been performed. Finally, the implementation uses *tabu lists* to store recently flipped variables, so as to avoid flipping them repeatedly.

Wu and Wah [1999] observed that in many difficult to solve instances, such as from the `parity` and `hanoi` domains, basic DLM frequently gets stuck in *traps*, i.e., pairs (x, λ) such that there are one or more unsatisfied clauses but the associated L_d increases by changing x in any one dimension. They found that on these instances, some clauses are significantly more likely than others to be amongst the unsatisfied clauses in such a trap. (Note that this is different from counting how often is a clause unsatisfied; here we only consider the clause status inside a trap situation, ignoring how the search arrived at the trap.) Keeping track of such clauses and periodically performing a special increase in their associated Lagrange multipliers, guides the search away from traps and results in better performance.

Wu and Wah [2000] later generalized this strategy by recording not only information about traps but a history of all recently visited points in the variable space. Instead of performing a special increase periodically, this history information is now added directly as a *distance penalty* term to the Lagrangian function

L_d. The penalty is larger for points that are in Hamming distance closer to the current value of x, thereby guiding the search away from recently visited points (in particular, points inside a trap that would be visited repeatedly).

6.3. The Phase Transition Phenomenon in Random k-SAT

One of the key motivations in the early 1990's for studying incomplete, stochastic methods for solving SAT problems was the observation that DPLL-based systematic solvers perform quite poorly on certain randomly generated formulas. For completeness, we provide a brief overview of these issues here; for a detailed discussion, refer to Part 1, Chapter 8 of this Handbook.

Consider a random k-CNF formula F on n variables generated by independently creating m clauses as follows: for each clause, select k distinct variables uniformly at random out of the n variables and negate each variable with probability 0.5. When F is chosen from this distribution, Mitchell, Selman, and Levesque [1992] observed that the median hardness of the problems is very nicely characterized by a key parameter: the *clause-to-variable ratio*, m/n, typically denoted by α. They observed that problem hardness peaks in a critically constrained region determined by α alone. The left pane of Figure 6.1 depicts the now well-known "easy-hard-easy" pattern of SAT and other combinatorial problems, as the key parameter (in this case α) is varied. For random 3-SAT, this region has been experimentally shown to be around $\alpha \approx 4.26$ (for early results see Crawford and Auton [1993], Kirkpatrick and Selman [1994]; new findings by Mertens et al. [2006]), and has provided challenging benchmarks as a test-bed for SAT solvers. Cheeseman et al. [1991] observed a similar easy-hard-easy pattern in random graph coloring problems. For random formulas, interestingly, a slight natural variant of the above "fixed-clause-length" model, called the variable-clause-length model, does *not* have a clear set of parameters that leads to a hard set of instances [Franco and Paull, 1983, Goldberg, 1979, Purdom Jr. and Brown, 1987]. This apparent difficulty in generating computationally hard instances for SAT solvers provided the impetus for much of the early work on local search methods for SAT. We refer the reader to Cook and Mitchell [1997] for a detailed survey.

The critically constrained region marks a stark transition not only in the computational hardness of random SAT instances but also in their satisfiability itself. The right pane of Figure 6.1 shows the fraction of random formulas that are unsatisfiable, as a function of α. We see that nearly all problems with α below the critical region (the under-constrained problems) are satisfiable. As α approaches and passes the critical region, there is a sudden change and nearly all problems in this over-constrained region are unsatisfiable. Further, as n grows, this phase transition phenomenon becomes sharper and sharper, and coincides with the region in which the computational hardness peaks. The relative hardness of the instances in the unsatisfiable region to the right of the phase transition is consistent with the formal result of Chvátal and Szemerédi [1988] who, building upon the work of Haken [1985], proved that large unsatisfiable random k-CNF formulas almost surely require exponential size resolution refutations, and thus exponential length runs of any DPLL-based algorithm proving unsatisfiability. This formal re-

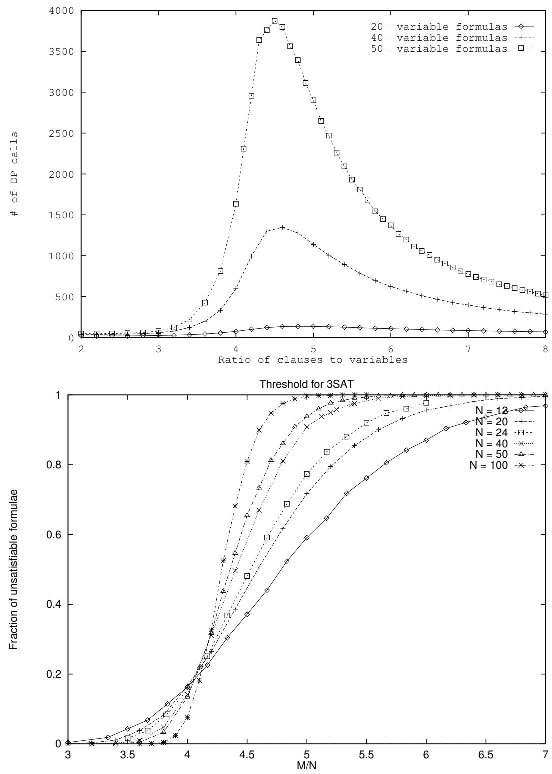

Figure 6.1. The phase transition phenomenon in random 3-SAT. Top: Computational hardness peaks at $\alpha \approx 4.26$. Bottom: Problems change from being mostly satisfiable to mostly unsatisfiable. The transitions sharpen as the number of variables grows.

sult was subsequently refined and strengthened by others [cf. Beame et al., 1998, Beame and Pitassi, 1996, Clegg et al., 1996].

Relating the phase transition phenomenon for 3-SAT to statistical physics, Kirkpatrick and Selman [1994] showed that the threshold has characteristics typical of phase transitions in the statistical mechanics of disordered materials (see also Monasson et al. [1999] and Part 2, Chapter 4 of this Handbook). Physicists have studied phase transition phenomena in great detail because of the many interesting changes in a system's macroscopic behavior that occur at phase boundaries. One useful tool for the analysis of phase transition phenomena is called *finite-size scaling* analysis. This approach is based on rescaling the horizontal axis by a factor that is a function of n. The function is such that the horizontal axis is stretched out for larger n. In effect, rescaling "slows down" the phase-transition for higher values of n, and thus gives us a better look inside the transition. From the resulting universal curve, applying the scaling function backwards, the actual transition curve for each value of n can be obtained. In principle, this approach also localizes the 50%-satisfiable-point for any value of

n, which allows one to generate very hard random 3-SAT instances.

Interestingly, it is still not formally known whether there even exists a critical constant α_c such that as n grows, almost all 3-SAT formulas with $\alpha < \alpha_c$ are satisfiable and almost all 3-SAT formulas with $\alpha > \alpha_c$ are unsatisfiable. In this respect, Friedgut [1999] provided the first positive result, showing that there exists a *function* $\alpha_c(n)$ depending on n such that the above threshold property holds. (It is quite likely that the threshold in fact does not depend on n, and is a fixed constant.) In a series of papers, researchers have narrowed down the gap between upper bounds on the threshold for 3-SAT [e.g. Broder et al., 1993, Dubois et al., 2000, Franco and Paull, 1983, Janson et al., 2000, Kirousis et al., 1996], the best so far being 4.596, and lower bounds [e.g. Achlioptas, 2000, Achlioptas and Sorkin, 2000, Broder et al., 1993, Franco, 1983, Frieze and Suen, 1996, Hajiaghayi and Sorkin, 2003, Kaporis et al., 2006], the best so far being 3.52. On the other hand, for random 2-SAT, we do have a full rigorous understanding of the phase transition, which occurs at clause-to-variable ratio of 1 [Bollobás et al., 2001, Chvátal and Reed, 1992]. Also, for general k, the threshold for random k-SAT is known to be in the range $2^k \ln 2 - \mathrm{O}(k)$ [Achlioptas et al., 2005, Gomes and Selman, 2005].

6.4. A New Technique for Random *k*-SAT: Survey Propagation

We end this section with a brief discussion of Survey Propagation (SP), an exciting new incomplete algorithm for solving hard combinatorial problems. The reader is referred to Part 2, Chapter 4 of this Handbook for a detailed treatment of work in this direction. Survey propagation was discovered in 2002 by Mézard, Parisi, and Zecchina [2002], and is so far the only known method successful at solving random 3-SAT instances with one million variables and beyond in near-linear time in the most critically constrained region.[3]

The SP method is quite radical in that it tries to approximate, using an iterative process of local "message" updates, certain marginal probabilities related to the set of satisfying assignments. It then assigns values to variables with the most extreme probabilities, simplifies the formula, and repeats the process. This strategy is referred to as SP-inspired decimation. In effect, the algorithm behaves like the usual DPLL-based methods, which also assign variable values incrementally in an attempt to find a satisfying assignment. However, quite surprisingly, SP almost never has to backtrack. In other words, the "heuristic guidance" from SP is almost always correct. Note that, interestingly, computing marginals on satisfying assignments is strongly believed to be much harder than finding a single satisfying assignment (#P-complete vs. NP-complete). Nonetheless, SP is able to efficiently approximate certain marginals on random SAT instances and uses this information to successfully find a satisfying assignment.

SP was derived from rather complex statistical physics methods, specifically, the so-called *cavity method* developed for the study of spin glasses. The origin

[3] As mentioned earlier, it has been recently shown that by finely tuning the noise and temperature parameters, `Walksat` can also be made to scale well on hard random 3-SAT instances with clause-to-variable ratios $\alpha > 4.2$ [Seitz et al., 2005].

of SP in statistical physics and its remarkable and unparalleled success on extremely challenging random 3-SAT instances has sparked a lot of interest in the computer science research community, and has led to a series of papers in the last five years exploring various aspects of SP [e.g. Achlioptas and Ricci-Tersenghi, 2006, Aurell et al., 2004, Braunstein and Zecchina, 2004, Kroc et al., 2007, 2008, Krzakala et al., 2007, Maneva, 2006, Maneva and Sinclair, 2007, Maneva et al., 2007, Mézard et al., 2005, Mézard and Zecchina, 2002, Zdeborova and Krzakala, 2007]. Many of these aspects still remain somewhat mysterious, making SP an active and promising research area for statistical physicists, theoretical computer scientists, and artificial intelligence practitioners alike.

While the method is still far from well-understood, close connections to belief propagation (BP) methods [Pearl, 1988] more familiar to computer scientists have been subsequently discovered. In particular, it was shown by Braunstein and Zecchina [2004] (later extended by Maneva, Mossel, and Wainwright [2007]) that SP equations are equivalent to BP equations for obtaining marginals over a special class of combinatorial objects, called covers. In this respect, SP is the first successful example of the use of a probabilistic reasoning technique to solve a purely combinatorial search problem. The recent work of Kroc et al. [2007] empirically established that SP, despite the very loopy nature of random formulas which violate the standard tree-structure assumptions underlying the BP algorithm, is remarkably good at computing marginals over these covers objects on large random 3-SAT instances. Kroc et al. [2008] also demonstrated that information obtained from BP-style algorithms can be effectively used to enhance the performance of algorithms for the model counting problem, a generalization of the SAT problem where one is interested in counting the number of satisfying assignments.

Unfortunately, the success of SP is currently limited to random SAT instances. It is an exciting research challenge to further understand SP and apply it successfully to more structured, real-world problem instances.

6.5. Conclusion

Incomplete algorithms for satisfiability testing provide a complementary approach to complete methods, using an essentially disjoint set of techniques and being often well-suited to problem domains in which complete methods do not scale well. While a mixture of greedy descent and random walk provide the basis for most local search SAT solvers in use today, much work has gone into finding the right balance and in developing techniques to focus the search and efficiently bring it out of local minima and traps. Formalisms like the discrete Lagrangian method and ideas like clause weighting or flooding have played a crucial role in pushing the understanding and scalability of local search methods for SAT. An important role has also been played by the random k-SAT problem, particularly in providing hard benchmarks and a connection to the statistical physics community, leading to the survey propagation algorithm. Can we bring together ideas and techniques from systematic solvers and incomplete solvers to create a solver that has the best of both worlds? While some progress has been made in this direction, much remains to be done.

Bibliography

D. Achlioptas. Setting 2 variables at a time yields a new lower bound for random 3-SAT. In *32nd STOC*, pages 28–37, Portland,OR, May 2000.

D. Achlioptas, A. Naor, and Y. Peres. Rigorous location of phase transitions in hard optimization problems. *Nature*, 435:759–764, 2005.

D. Achlioptas and F. Ricci-Tersenghi. On the solution-space geometry of random constraint satisfaction problems. In *38th STOC*, pages 130–139, Seattle, WA, May 2006.

D. Achlioptas and G. Sorkin. Optimal myopic algorithms for random 3-SAT. In *41st FOCS*, pages 590–600, Redondo Beach, CA, Nov. 2000. IEEE.

H. Adorf and M. Johnston. A discrete stochastic neural network algorithm for constraint satisfaction problems. In *Intl. Joint Conf. on Neural Networks*, pages 917–924, San Diego, CA, 1990.

Anbulagan, D. N. Pham, J. K. Slaney, and A. Sattar. Old resolution meets modern SLS. In *20th AAAI*, pages 354–359, Pittsburgh, PA, July 2005.

E. Aurell, U. Gordon, and S. Kirkpatrick. Comparing beliefs, surveys, and random walks. In *17th NIPS*, Vancouver, BC, Dec. 2004.

P. Beame, R. Karp, T. Pitassi, and M. Saks. On the Complexity of Unsatisfiability Proofs for Random k-CNF Formulas. In *30th STOC*, pages 561–571, Dallas, TX, May 1998.

P. W. Beame and T. Pitassi. Simplified and improved resolution lower bounds. In *37th FOCS*, pages 274–282, Burlington, VT, Oct. 1996. IEEE.

B. Bollobás, C. Borgs, J. T. Chayes, J. H. Kim, and D. B. Wilson. The scaling window of the 2-SAT transition. *Random Struct. and Alg.*, 19(3-4):201–256, 2001.

A. Braunstein and R. Zecchina. Survey propagation as local equilibrium equations. *J. Stat. Mech.*, P06007, 2004. URL `http://lanl.arXiv.org/cond-mat/0312483`.

A. Broder, A. Frieze, and E. Upfal. On the satisfiability and maximum satisfiability of random 3-CNF formulas. In *Proc., 4th SODA*, Jan. 1993.

B. Cha and K. Iwama. Adding new clauses for faster local search. In *13th AAAI*, pages 332–337, Portland, OR, Aug. 1996.

P. Cheeseman, B. Kenefsky, and W. Taylor. Where the really hard problems are. In *Proceedings of IJCAI-91*, pages 331–337. Morgan Kaufmann, 1991.

V. Chvátal and B. Reed. Mick gets some (the odds are on his side). In *33rd FOCS*, pages 620–627, Pittsburgh, PA, Oct. 1992. IEEE.

V. Chvátal and E. Szemerédi. Many hard examples for resolution. *J. Assoc. Comput. Mach.*, 35(4):759–768, 1988.

M. Clegg, J. Edmonds, and R. Impagliazzo. Using the Gröbner basis algorithm to find proofs of unsatisfiability. In *28th STOC*, pages 174–183, Philadelphia, PA, May 1996.

S. Cook and D. Mitchell. Finding hard instances of the satisfiability problem: a survey. In *DIMACS Series in Discr. Math. and Theoretical Comp. Sci.*, volume 35, pages 1–17. American Math. Society, 1997.

J. Crawford and L. Auton. Experimental results on the cross-over point in satisfiability problems. In *Proc. AAAI-93*, pages 21–27, Washington, DC, 1993.

E. Dantsin, A. Goerdt, E. A. Hirsch, R. Kannan, J. M. Kleinberg, C. H. Papadimitriou, P. Raghavan, and U. Schöning. A deterministic $(2 - 2/(k+1))^n$ algorithm for k-SAT based on local search. *Theoretical Comput. Sci.*, 289(1): 69–83, 2002.

O. Dubois, Y. Boufkhad, and J. Mandler. Typical random 3-SAT formulae and the satisfiability threshold. In *Proc., 11th SODA*, pages 126–127, San Francisco, CA, Jan. 2000.

J. Franco. Probabilistic analysis of the pure literal heuristic for the satisfiability problem. *Annals of Operations Research*, 1:273–289, 1983.

J. Franco and M. Paull. Probabilistic analysis of the Davis-Putnam procedure for solving the satisfiability problem. *Discr. Applied Mathematics*, 5:77–87, 1983.

J. Frank. Learning short-term weights for GSAT. In *15th IJCAI*, pages 384–391, Nagoya, Japan, Aug. 1997.

J. Frank, P. Cheeseman, and J. Stutz. Where gravity fails: Local search topology. *JAIR*, 7:249–281, 1997.

E. Friedgut. Sharp thresholds of graph properties, and the k-sat problem. *Journal of the American Mathematical Society*, 12:1017–1054, 1999.

A. Frieze and S. Suen. Analysis of two simple heuristics on a random instance of k-SAT. *J. Alg.*, 20(2):312–355, 1996.

I. P. Gent and T. Walsh. Towards an understanding of hill-climbing procedures for SAT. In *11th AAAI*, pages 28–33, Washington, DC, July 1993.

M. L. Ginsberg and D. A. McAllester. GSAT and dynamic backtracking. In *4th KR*, pages 226–237, Bonn, Germany, May 1994.

A. Goldberg. On the complexity of the satisfiability problem. Technical Report Report No. 16, Courant Computer Science, New York University, 1979.

C. P. Gomes and B. Selman. Can get satisfaction. *Nature*, 435:751–752, 2005.

J. Gu. Efficient local search for very large-scale satisfiability problems. *Sigart Bulletin*, 3(1):8–12, 1992.

J. Gu, P. W. Purdom, J. Franco, and B. J. Wah. Algorithms for the Satisfiability (SAT) Problem: A Survey. In *Satisfiability (SAT) Problem*, DIMACS, pages 19–151. American Mathematical Society, 1997.

D. Habet, C. M. Li, L. Devendeville, and M. Vasquez. A hybrid approach for SAT. In *8th CP*, volume 2470 of *LNCS*, pages 172–184, Ithaca, NY, Sept. 2002.

M. Hajiaghayi and G. Sorkin. The satisfiability threshold for random 3-SAT is at least 3.52, 2003. URL `http://arxiv.org/abs/math/0310193`.

A. Haken. The intractability of resolution. *Theoretical Comput. Sci.*, 39:297–305, 1985.

M. Heule and H. van Maaren. March_dl: Adding adaptive heuristics and a new branching strategy. *J. Satisfiability*, 2:47–59, 2006.

M. Heule, J. van Zwieten, M. Dufour, and H. van Maaren. March_eq: Implementing additional reasoning into an efficient lookahead SAT solver. In *7th SAT*, volume 3542 of *LNCS*, pages 345–359, Vancouver, BC, May 2004.

E. A. Hirsch and A. Kojevnikov. UnitWalk: A new SAT solver that uses local search guided by unit clause elimination. *Annals Math. and AI*, 43(1):91–111, 2005.

J. N. Hooker. A quantitative approach to logical inference. *Decision Support Systems*, 4:45–69, 1988.

H. Hoos. On the run-time behaviour of stochastic local search algorithms for SAT. In *Proceedings of AAAI-99*, pages 661–666. AAAI Press, 1999.

H. H. Hoos. An adaptive noise mechanism for WalkSAT. In *18th AAAI*, pages 655–660, Edmonton, Canada, July 2002.

H. H. Hoos and T. Stützle. *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann, San Francisco, CA, 2004.

F. Hutter, D. A. D. Tompkins, and H. H. Hoos. Scaling and probabilistic smoothing: Efficient dynamic local search for SAT. In *8th CP*, volume 2470 of *LNCS*, pages 233–248, Ithaca, NY, Sept. 2002.

S. Janson, Y. C. Stamatiou, and M. Vamvakari. Bounding the unsatisfiability threshold of random 3-SAT. *Random Struct. and Alg.*, 17(2):103–116, 2000.

D. Johnson, C. Aragon, L. McGeoch, and C. Schevon. Optimization by simulated annealing: an experimental evaluation; part II. *Operations Research*, 39, 1991.

D. S. Johnson and M. A. Trick, editors. *Cliques, Coloring and Satisfiability: the Second DIMACS Implementation Challenge*, volume 26 of *DIMACS Series in DMTCS*. Amer. Math. Soc., 1996.

A. Kamath, N. Karmarkar, K. Ramakrishnan, and M. Resende. Computational experience with an interior point algorithm on the satisfiability problem. In *Proceedings of Integer Programming and Combinatorial Optimization*, pages 333–349, Waterloo, Canada, 1990. Mathematical Programming Society.

A. C. Kaporis, L. M. Kirousis, and E. G. Lalas. The probabilistic analysis of a greedy satisfiability algorithm. *Random Structures and Algorithms*, 28(4): 444–480, 2006.

S. Kirkpatrick, D. Gelatt Jr., and M. Vecchi. Optimization by simuleated annealing. *Science*, 220(4598):671–680, 1983.

S. Kirkpatrick and B. Selman. Critical behavior in the satisfiability of random Boolean expressions. *Science*, 264:1297–1301, May 1994. Also p. 1249: "Math. Logic: Pinning Down a Trecherous Border in Logical Statements" by B. Cipra.

L. M. Kirousis, E. Kranakis, and D. Krizanc. Approximating the unsatisfiability threshold of random formulas. In *Proceedings of the Fourth Annual European Symposium on Algorithms*, pages 27–38, Barcelona, Spain, Sept. 1996.

K. Konolige. Easy to be hard: Difficult problems for greedy algorithms. In *4th KR*, pages 374–378, Bonn, Germany, May 1994.

L. Kroc, A. Sabharwal, and B. Selman. Survey propagation revisited. In *23rd UAI*, pages 217–226, Vancouver, BC, July 2007.

L. Kroc, A. Sabharwal, and B. Selman. Leveraging belief propagation, backtrack search, and statistics for model counting. In *5th CPAIOR*, volume 5015 of

LNCS, pages 127–141, Paris, France, May 2008.

F. Krzakala, A. Montanari, F. Ricci-Tersenghi, G. Semerjian, and L. Zdeborova. Gibbs states and the set of solutions of random constraint satisfaction problems. *PNAS*, 104(25):10318–10323, June 2007.

C. M. Li, W. Wei, and H. Zhang. Combining adaptive noise and look-ahead in local search for SAT. In *10th SAT*, volume 4501 of *LNCS*, pages 121–133, Lisbon, Portugal, May 2007.

X. Y. Li, M. F. M. Stallmann, and F. Brglez. QingTing: A local search SAT solver using an effective switching strategy and an efficient unit propagation. In *6th SAT*, pages 53–68, Santa Margherita, Italy, May 2003.

S. Lin and B. Kernighan. An efficient heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21:498–516, 1973.

E. Maneva. *Belief Propagation Algorithms for Constraint Satisfaction Problems*. PhD thesis, University of California, Berkeley, 2006.

E. Maneva, E. Mossel, and M. J. Wainwright. A new look at survey propagation and its generalizations. *J. Assoc. Comput. Mach.*, 54(4):17, July 2007.

E. Maneva and A. Sinclair. On the satisfiability threshold and survey propagation for random 3-SAT, 2007. http://arxiv.org/abs/cs.CC/0609072.

B. Mazure, L. Sais, and E. Gregoire. Boosting complete techniques thanks to local search methods. In *Proc. Math and AI*, 1996.

B. Mazure, L. Sais, and E. Grégoire. Tabu search for SAT. In *14th AAAI*, pages 281–285, Providence, RI, July 1997.

D. A. McAllester, B. Selman, and H. Kautz. Evidence for invariants in local search. In *AAAI/IAAI*, pages 321–326, Providence, RI, July 1997.

S. Mertens, M. Mézard, and R. Zecchina. Threshold values of random K-SAT from the cavity method. *Random Struct. and Alg.*, 28(3):340–373, 2006.

M. Mézard, T. Mora, and R. Zecchina. Clustering of solutions in the random satisfiability problem. *Phy. Rev. Lett.*, 94:197205, May 2005.

M. Mézard, G. Parisi, and R. Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297(5582):812–815, 2002.

M. Mézard and R. Zecchina. Random k-satisfiability problem: From an analytic solution to an efficient algorithm. *Phy. Rev. E*, 66:056126, Nov. 2002.

S. Minton, M. Johnston, A. Philips, and P. Laird. Solving large-scale constraint satisfaction an scheduling problems using a heuristic repair method. In *Proceedings AAAI-90*, pages 17–24. AAAI Press, 1990.

D. Mitchell, B. Selman, and H. Levesque. Hard and easy distributions of SAT problems. In *Proc. AAAI-92*, pages 459–465, San Jose, CA, 1992.

R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky. Determining computational complexity from characteristic phase transitions. *Nature*, 400(8):133–137, 1999.

P. Morris. The breakout method for escaping from local minima. In *11th AAAI*, pages 428–433, Washington, DC, July 1993.

C. Papadimitriou and K. Steiglitz. *Combinatorial Optimization*. Prentice-Hall, Inc., 1982.

C. H. Papadimitriou. On selecting a satisfying truth assignment. In *32nd FOCS*, pages 163–169, San Juan, Puerto Rico, Oct. 1991. IEEE.

A. J. Parkes and J. P. Walser. Tuning local search for satisfiability testing. In

13th AAAI, pages 356–362, Portland, OR, Aug. 1996.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

D. N. Pham, J. Thornton, and A. Sattar. Building structure into local search for SAT. In *20th IJCAI*, pages 2359–2364, Hyderabad, India, Jan. 2007.

S. D. Prestwich. Local search and backtracking vs non-systematic backtracking. In *AAAI 2001 Fall Symp. on Using Uncertainty within Computation*, 2001.

S. D. Prestwich. Variable dependency in local search: Prevention is better than cure. In *10th SAT*, Lisbon, Portugal, May 2007.

P. W. Purdom Jr. and C. A. Brown. Polynomial average-time satisfiability problems. *Information Science*, 41:23–42, 1987.

M. G. C. Resende and T. A. Feo. A GRASP for satisfiability. In Johnson and Trick [1996], pages 499–520.

I. Rish and R. Dechter. To guess or to think? hybrid algorithms for SAT. In *Proc. Conference on Principles of Constraint Programming (CP-96)*, pages 555–556, 1996.

U. Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *40th FOCS*, pages 410–414, New York, NY, Oct. 1999. IEEE.

U. Schöning. A probabilistic algorithm for k-SAT based on limited local search and restart. *Algorithmica*, 32(4):615–623, 2002.

D. Schuurmans and F. Southey. Local search characteristics of incomplete SAT procedures. *AI J.*, 132(2):121–150, 2001.

D. Schuurmans, F. Southey, and R. C. Holte. The exponentiated subgradient algorithm for heuristic boolean programming. In *17th IJCAI*, pages 334–341, Seattle, WA, Aug. 2001.

S. Seitz, M. Alava, and P. Orponen. Focused local search for random 3-satisfiability. *J. Stat. Mech.*, P06006, 2005.

B. Selman and H. Kautz. Domain-independent extensions to GSAT: Solving large structured satisfiability problems. In *13th IJCAI*, pages 290–295, France, 1993.

B. Selman, H. Kautz, and B. Cohen. Noise strategies for local search. In *Proc. AAAI-94*, pages 337–343, Seattle, WA, 1994.

B. Selman, H. Kautz, and B. Cohen. Local search strategies for satisfiability testing. In Johnson and Trick [1996], pages 521–532.

B. Selman, H. J. Levesque, and D. G. Mitchell. A new method for solving hard satisfiability problems. In *10th AAAI*, pages 440–446, San Jose, CA, July 1992.

Y. Shang and B. W. Wah. A discrete Lagrangian-based global-search method for solving satisfiability problems. *J. of Global Optimization*, 12(1):61–99, 1998.

W. M. Spears. Simulated annealing for hard satisfiability problems. In Johnson and Trick [1996], pages 533–558.

J. Thornton, D. N. Pham, S. Bain, and V. Ferreira Jr. Additive versus multiplicative clause weighting for SAT. In *19th AAAI*, pages 191–196, San Jose, CA, July 2004.

Z. Wu and B. W. Wah. Trap escaping strategies in discrete Lagrangian methods for solving hard satisfiability and maximum satisfiability problems. In *16th AAAI*, pages 673–678, Orlando, FL, July 1999.

Z. Wu and B. W. Wah. An efficient global-search strategy in discrete Lagrangian methods for solving hard satisfiability problems. In *17th AAAI*, pages 310–315, Austin, TX, July 2000.

L. Zdeborova and F. Krzakala. Phase transition in the coloring of random graphs. *Phy. Rev. E*, 76:031131, Sept. 2007.

Index