Reduced Cost Fixing for Maximum Satisfiability[*]

Fahiem Bacchus
University of Toronto
Canada

Antti Hyttinen
University of Helsinki
Finland

Matti Järvisalo
University of Helsinki
Finland

Paul Saikko
University of Helsinki
Finland

Abstract

Maximum satisfiability (MaxSAT) offers a competitive approach to solving NP-hard real-world optimization problems. While state-of-the-art MaxSAT solvers rely heavily on Boolean satisfiability (SAT) solvers, a recent trend, brought on by MaxSAT solvers implementing the so-called implicit hitting set (IHS) approach, is to integrate techniques from the realm of integer programming (IP) into the solving process. This allows for making use of additional IP solving techniques to further speed up MaxSAT solving. In this line of work, we investigate the integration of the technique of reduced cost fixing from the IP realm into IHS solvers, and empirically show that reduced cost fixing considerable speeds up a state-of-the-art MaxSAT solver implementing the IHS approach.

1 Introduction

Beyond its importance as a classical NP-hard optimization problem, maximum satisfiability (MaxSAT) is today a thriving constraint optimization paradigm, with successful applications in a range of real-world domains. The underlying declarative language of MaxSAT consists of conjunctive normal form (CNF) propositional formulas extended with weights assigned to individual clauses. The aim is to find an optimal truth assignment, i.e., one that maximizes the weight of satisfied clauses. Algorithmically, however, MaxSAT is typically treated as the equivalent problem of *minimizing* the weight (or cost) of falsified clauses.

The currently most successful exact MaxSAT solvers make iteratively use of Boolean satisfiability (SAT) solvers, either to extract unsatisfiable cores (unsatisfiable sets of soft clauses) or to extract truth assignments with ever-improving cost (e.g., [Koshimura *et al.*, 2012]). Pure SAT-based solvers that extract cores (e.g., [Martins *et al.*, 2014; Ansótegui *et al.*, 2015; Alviano *et al.*, 2015; Morgado *et al.*, 2014]) use

these cores to modify the formula until eventually a formula is obtained that is satisfiable and whose satisfying truth assignments are optimal truth assignments for the original input formula. Implicit hitting set (IHS) solvers [Davies and Bacchus, 2011; Davies, 2013; Saikko *et al.*, 2016a] also extract cores, but then use an integer programming (IP) solver to compute a minimum-cost hitting sets (MCHS) of the accumulated set of cores. If removing such a hitting set (which is a set of soft clauses) from the input formula makes the formula satisfiable, then the satisfying truth assignment must be an optimal truth assignment for the original input formula.

The hybrid approach implemented in IHS solvers for MaxSAT opens up opportunities for making use of various proven IP solving techniques to further speed up MaxSAT solving. In this work we investigate utilizing the known IP technique of *reduced cost fixing* for this purpose.

In the context of MaxSAT, truth assignments found during solving provide an upper bound on the cost of an optimal truth assignment. Every MCHS of the cores on the other hand provides a lower bound. The linear programming (LP) relaxation of the MCHS problem thus also provides a lower bound. This LP can also be used to obtain a reduced cost for every soft clause c_i. The reduced cost of c_i specifies a minimum increase in the cost of the LP that would arise from falsifying (satisfying) c_i which was satisfied (falsified) in the LP solution. If this cost increase makes the LP MCHS lower bound greater than the best known upper bound, we can conclude that c_i must be fixed to its status in the LP solution; that is, c_i must be falsified (satisfied) by any optimal truth assignment. This reasoning can be extended to cover the case where the increase in cost makes the lower bound equal to the upper bound. In this case, we know that fixing the status of c_i preserves at least one optimal solution. Fixing the status of various soft clauses makes solving MaxSAT instances easier, as our empirical results demonstrate.

To facilitate the integration of reduced cost fixing, we present IHS search in terms of improving upper and lower bounds. To study the impact of reduced cost fixing for MaxSAT, we present results from an extensive empirical evaluation on the effect of integrating reduced cost fixing into the MaxHS IHS solver, showing that reduced cost fixing considerable speeds up this state-of-the-art MaxSAT solver.

In terms of related work, different techniques of using lower and upper bounds for speeding up MaxSAT solvers

have been studied in varying contexts, including branch and bound for MaxSAT [Li *et al.*, 2006; Lin and Su, 2007; Lin *et al.*, 2008; Li *et al.*, 2008], use of bounds for MaxSAT solvers in general [Heras *et al.*, 2012], and hardening based on the costs of residual formulas in pure SAT-based core-guided MaxSAT solving [Ansótegui *et al.*, 2013; Morgado *et al.*, 2012]. To the best of our knowledge, linear programming relaxation based reduced cost fixing has not been previously proposed in the context of MaxSAT. However, besides being a standard technique in IP solving [Wolsey, 1998; Danzig *et al.*, 1954; Crowder *et al.*, 1983; Nemhauser and Wolsey, 1999], there have been a number of recent works on exploiting reduced cost fixing in constraint programming, IP/constraint logic programming, and IP/constraint programming, e.g., [Thorsteinsson and Ottosson, 2002; Focacci *et al.*, 1999; Yunes *et al.*, 2010].

2 Maximum Satisfiability and Hitting Sets

An instance of (weighted partial) MaxSAT (F_h, F_s, wt) consists a set of hard clauses F_h, a set of soft clauses F_s, and weight function $wt : F_s \rightarrow \mathbb{R}^+$. A MaxSAT solution τ is a truth assignment for the variables in $F_h \cup F_s$ that satisfies all clauses in F_h. The cost of a solution, $cost(\tau)$, is the sum of the weights of soft clauses not satisfied by τ. A MaxSAT solution τ' is optimal if it has minimum cost over all solutions, $\tau' \in \arg\min_\tau(cost(\tau))$. A subset of soft clauses $\kappa \subseteq F_s$ is an unsatisfiable core if the clauses in κ cannot be simultaneously satisfied by any solution.

A hitting set HS of a set of cores $\mathcal{K}$ is a set which contains at least one element of each $\kappa \in \mathcal{K}$, i.e., $HS \cap \kappa \neq \emptyset$ for all $\kappa \in \mathcal{K}$. A minimum-cost hitting set (MCHS) of $\mathcal{K}$ is a hitting set which minimizes $\sum_{c_i \in HS} wt(c_i)$ over all hitting sets HS of $\mathcal{K}$. The optimal cost of a MaxSAT instance is equal to the cost of the minimum-cost hitting sets of the set of *all* cores of the MaxSAT instance.

Example 2.1. *Consider the MaxSAT instance* (F_h, F_s, wt) *with the hard clauses*

$$F_h = \{ \ x_1 \lor x_2, \quad x_2 \lor x_3, \quad x_1 \lor x_4, \quad \neg x_4 \lor \neg x_5,$$
$$\neg x_1 \lor \neg x_6, \quad \neg x_2 \lor \neg x_7, \quad \neg x_3 \lor \neg x_8 \ \},$$

and soft clauses

$$F_s = \{ \ \underbrace{x_5}_{c_1}, \quad \underbrace{x_6}_{c_2}, \quad \underbrace{x_7}_{c_3}, \quad \underbrace{x_8}_{c_4} \ \}$$

with $wt(c_1) = wt(c_2) = 2$, $wt(c_3) = 4$, $wt(c_4) = 1$.

The cores of this instance form the set $\mathcal{K} = \{\{c_1, c_2\}, \{c_2, c_3\}, \{c_3, c_4\}\}$. *For example,* $\{c_1, c_2\}$ *is a core since* $x_5 = 1$ (c_1) *implies* $x_4 = 0$ *and* $x_1 = 1$, *and* $x_6 = 1$ (c_2) *implies* $x_1 = 0$; *thus both cannot be satisfied by the same solution. An optimal solution is* $\tau : x_1 = x_3 = x_5 = x_7 = 1$, $x_2 = x_4 = x_6 = x_8 = 0$ *with* $cost(\tau) = 3$. *One MHCS of* $\mathcal{K}$ *is* $\{c_1, c_3\}$ *with cost* 3.

3 The Implicit Hitting Set Approach

We focus on IHS MaxSAT solvers [Davies and Bacchus, 2011; 2013a; 2013b; Davies, 2013; Saikko *et al.*, 2016a] which instantiate the implicit hitting set paradigm [Karp,
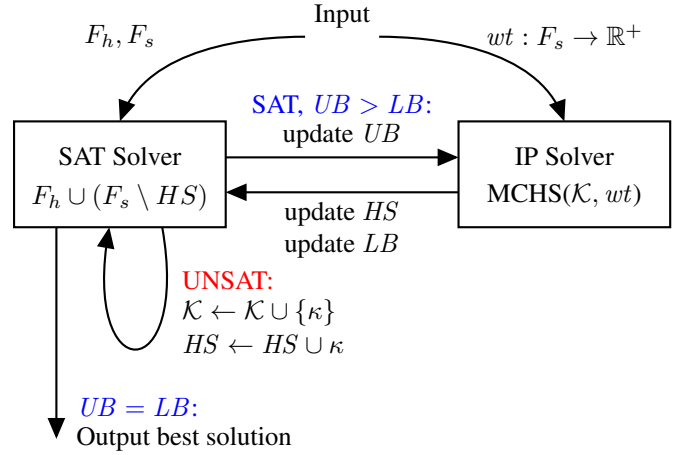

Figure 1: A bounds-based view on IHS for MaxSAT

2010; Moreno-Centeno and Karp, 2013; Saikko *et al.*, 2016b] as natural candidates for integrating reduced cost fixing. IHS solvers iteratively alternate between computing a set of unsatisfiable cores $\mathcal{K}$ of the MaxSAT instance and minimum-cost hitting sets over the mononotonically growing $\mathcal{K}$. To obtain a MCHS HS, an IP solver is employed on the MCHS IP formulation

$$\text{minimize} \quad \sum_{c_i \in F_s} wt(c_i) \cdot b_i$$
$$\text{subject to} \quad \sum_{c_i \in \kappa} b_i \geq 1 \qquad \forall \kappa \in \mathcal{K},$$
$$b_i \in \{0, 1\} \qquad \forall c_i \in F_s,$$

associating a new binary variable b_i with each soft clause c_i. At each iteration the satisfiability of $F_h \cup (F_s \setminus HS)$ is checked with a SAT solver. If satisfiable, the set HS implicitly hits *all* cores of the instance and, as first shown in [Davies and Bacchus, 2011], the optimality of HS implies the optimality of the MaxSAT solution, terminating the MaxSAT search.

To make use of reduced cost fixing, upper bounds on the cost of optimal MaxSAT solutions are needed. As originally described in [Davies and Bacchus, 2011], IHS does not directly provide upper bounds during search. However, upper bounds are obtained in the IHS solvers MaxHS and LMHS when non-optimal hitting sets are used as described in [Davies and Bacchus, 2013b]. In particular, for any hitting set HS, if the SAT solver finds a satisfying assignment τ for $F_h \cup (F_s \setminus HS)$, then $cost(\tau)$ is an upper bound ($cost(\tau)$ is equal to the optimal cost if HS is a MCHS).

Here we provide a bounds-based view on the iterations and termination of the IHS algorithm for MaxSAT. Illustrated in Figure 1, the bounds-based IHS algorithm for MaxSAT starts by initializing the SAT solver and IP solver with their respective inputs. The SAT solver is initialized with the CNF formula $F_h \cup F_s$, while the IP solver is initialized with the soft clause weights wt. The hitting set HS and set of cores $\mathcal{K}$ are both initially empty. The upper bound UB is initialized to ∞ and LB to 0. We assume for simplicity that F_h is satisfiable. The first SAT solver invocation tests the entire set of clauses

$F_h \cup F_s$. If the formula is satisfiable, we obtain an upper bound of 0 and terminate as $LB = UB$. Otherwise we obtain an unsatisfiable core κ and add it to $\mathcal{K}$.

In order to obtain the upper bounds required for reduced cost fixing, we deviate from the original IHS algorithm. Instead of immediately computing a new optimal hitting set for $\mathcal{K}$ we add the clauses of the new core κ to HS, forming a *non-optimal hitting set* of $\mathcal{K}$, and check the satisfiability of $F_h \cup (F_s \setminus HS)$ again. (A non-optimal hitting set of $\mathcal{K} \cup \{\kappa\}$ could be also obtained in other ways [Davies and Bacchus, 2013b]). This is repeated, adding cores κ to $\mathcal{K}$ and the clauses of the new core to HS, until $F_h \cup (F_s \setminus HS)$ becomes satisfiable (this is the "disjoint" strategy of [Saikko, 2015]). F_h is satisfiable, so this condition will eventually be met when HS grows sufficiently large. Once $F_h \cup (F_s \setminus HS)$ is satisfiable, a MaxSAT solution is obtained. Its cost gives an upper bound value and UB is updated. At this point the IP solver is invoked to compute a minimum-cost hitting set HS for $\mathcal{K}$. The hitting set HS is optimal, so its cost gives a lower bound for the cost of the optimal MaxSAT solution and LB is updated.

The algorithm terminates when $LB = UB$ and returns the MaxSAT solution which yielded UB. By the same argument as presented in [Davies and Bacchus, 2011] the algorithm must eventually terminate. A detailed proof of correctness is provided in [Bacchus *et al.*, 2017].

Example 3.1. *Consider again the MaxSAT instance (F_h, F_s, wt) from Example 2.1. Following Figure 1, a SAT solver is called on $F_h \cup F_s$. Assume that it returns UNSAT with the core $\kappa = \{c_2, c_3\}$. The SAT solver is called again on $F_h \cup (F_s \setminus \kappa)$, now returning the solution $\tau : x_1 = x_3 = x_5 = x_7 = 1$, $x_2 = x_4 = x_6 = x_8 = 0$. This solution has $cost(\tau) = 3$, which gives an upper bound $UB = 3$. The IP solver then called to compute a minimum-cost hitting set over $\mathcal{K} = \{\{c_2, c_3\}\}$. Clearly, $HS = \{c_2\}$. The SAT solver is called on $(F_h \cup F_s) \setminus \{c_2\}$. It returns UNSAT with core $\kappa = \{c_3, c_4\}$. The next SAT solver call over $(F_h \cup F_s) \setminus \{c_2, c_3, c_4\}$ returns SAT but cannot improve UB. The IP solver is called for a minimum-cost hitting set over $\mathcal{K} = \{\{c_2, c_3\}, \{c_3, c_4\}\}$. This gives $HS = \{c_2, c_4\}$ and $LB = 3$. Now $UB = LB$ and the search terminates.*

4 Reduced Cost Fixing for IHS

Reduced cost fixing is a standard technique in IP solving [Wolsey, 1998; Danzig *et al.*, 1954; Crowder *et al.*, 1983; Nemhauser and Wolsey, 1999]. It uses an upper bound and reduced costs obtained from an LP relaxation to fix variables in an IP. Given a minimization IP P containing Boolean (0/1) variables, we can solve P as an LP by allowing the Boolean variables to take on intermediate values between 0 and 1. The cost of the LP solution will be a lower bound on the optimal cost of P. The LP solver also provides a reduced cost for the non-basic[1] variables set at 0 or 1 in the LP solution. These reduced costs specify a minimum increase in the cost of the LP that would arise from changing a non-basic variable at 0 (1) to 1 (0). Suppose we know a feasible IP solution to P with

---

[1]The variables in the LP solution are either basic or non-basic. All of the non-basic variables will be at their upper or lower bounds in the LP solution [Chvátal, 1983].

cost z. If changing a non-basic variable causes the LP solution to increase in cost beyond z, then we can fix that variable to the value it has in the LP solution. Since the LP solution is a lower bound, putting such variables at their opposite values would cause the cost of the IP to increase beyond the cost of an already known feasible solution. Here we explain how this technique can be used within IHS MaxSAT solvers.

Theorem 4.1. *For a MaxSAT problem $F = (F_h, F_s, wt)$, suppose we have (a) $B = \{b_1, \ldots, b_n\}$ a set of Boolean variables where each $b_i = 0$ ($b_i = 1$) represents the satisfaction (falsification) of a soft clause $c_i \in F_s$, (b) IP_{HS} an IP over the b_i representing the minimum-cost hitting set problem over the current set of cores, (c) LP_{HS} the LP relaxation of IP_{HS}, (d) $best\_\tau$ a feasible solution to F, (e) an optimal solution to LP_{HS} with cost $z_{opt}^{LP_{HS}}$, and (f) LP reduced costs $rc(b_i)$ at the optimal basis.*

Then the following simplifications can be performed without changing $opt\_cost(F)$. (1) For every non-basic variable b_i set to 0 in the optimal LP_{HS} solution we can make the soft clause c_i hard in F if $z_{opt}^{LP_{HS}} + rc(b_i) > cost(best\_\tau)$ or if $z_{opt}^{LP_{HS}} + rc(b_i) = cost(best\_\tau)$ and c_i is satisfied in $best\_\tau$. (2) For every non-basic variable b_i set to 1 in the optimal LP_{HS} we can make soft clause c_i false in F if $z_{opt}^{LP_{HS}} - rc(b_i) > cost(best\_\tau)$ or if $z_{opt}^{LP_{HS}} - rc(b_i) = cost(best\_\tau)$ and c_i is falsified in $best\_\tau$.

For more intuition on how reduced cost fixing allows for hardening and falsifying soft clauses during IHS search for an optimal MaxSAT solution, we consider the following example.

Example 4.2. *Consider again the MaxSAT instance (F_h, F_s, wt) from Example 2.1 and the first iteration of the execution of IHS as described in Example 3.1. After obtaining from the SAT solver the first solution $best\_\tau$ with $cost(best\_\tau) = 3$, the IP solver is called for a hitting set over $\mathcal{K} = \{\{c_2, c_3\}\}$. The LP-relaxation of the hitting set problem LP_{HS} can be formulated as*

$$\text{minimize } 2b_1 + 2b_2 + 4b_3 + 1b_4$$
$$\text{subject to } b_2 + b_3 - s = 1,$$
$$0 \leq s, 0 \leq b_1, b_2, b_3, b_4 \leq 1,$$

where s is a surplus variable. This LP can be solved using different algorithms, clearly the optimal LP solution is $b_2 = 1, b_1 = b_3 = b_4 = s = 0$, which has cost $z_{opt}^{LP_{HS}} = 2$. This induces a lower bound $LB = 2$. In this case, the optimal LP solution happens to be the optimal solution to the IP HS problem, i.e. $HS = \{c_2\}$.

A feasible solution to an LP can be represented using a basis, where non-basic variables are at their upper or lower bounds. Consider the basis consisting of b_3. The basic variables can be written as a function of the non-basic ones, in this case as $b_3 = 1 - b_2 + s$. This allows the objective to be rewritten solely as a function of the non-basic variables as

$$2b_1 + 2b_2 + 4\underbrace{(1 - b_2 + s)}_{b_3} + b_4 = 4 + 2b_1 - 2b_2 + b_4 + 4s$$

The coefficients in front of the non-basic variables are the reduced costs *and (obviously) specify the change of the objective to modifications of the non-basic variables. Now, since*
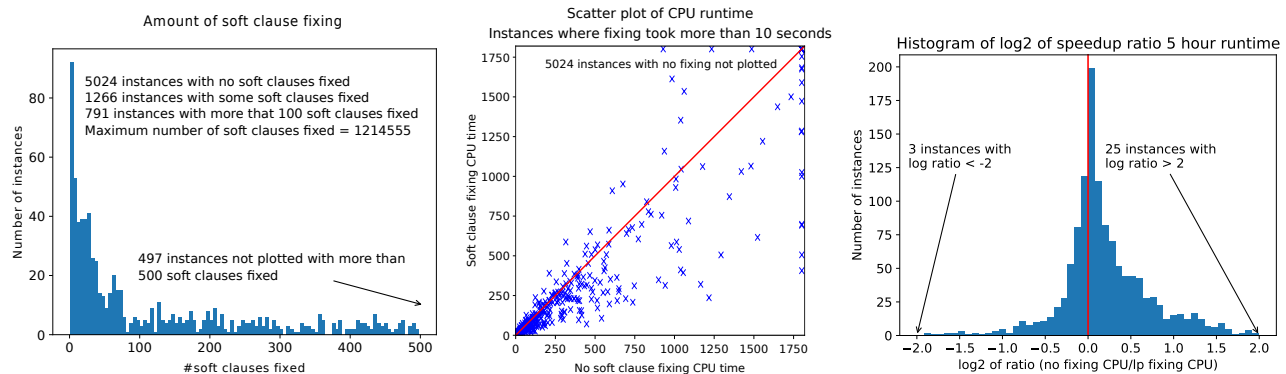
Figure 2: Left: distribution of the frequency reduced cost fixing forces a soft clause to be relaxed or hardened. Middle: Scatter plot showing that fixing on instances where fixing takes significant time also pays off. Right: Speedup histogram under 5-hour per-instance time limit over instances on which reduced cost fixing would force some variables in terms of $\log_2$ of CPU time with fixing and without fixing.

b_1 is set to 0 in the optimal LP solution and $z_{opt}^{LP_{HS}} + rc(b_1) = 2 + 2 > 3 = cost(best\_\tau)$, we can make c_1 hard. Intuitively, solutions under which c_1 is falsified ($b_1 = 1$) would have cost at least 4 and hence such solutions cannot be optimal for the MaxSAT instance. Since b_2 is set to 1 in the LP solution and $z_{opt}^{LP_{HS}} - rc(b_2) = 2 - (-2) > 3 = cost(best\_\tau)$, we can make c_2 false. Intuitively, solutions under which c_2 is satisfied ($b_2 = 0$) would have cost at least 4 and hence such solutions cannot be optimal, either.

5 Empirical Results

We implemented reduced cost fixing in MaxHS v2.9.8 which entered the 2016 MaxSAT Evaluation. This version of MaxHS includes a number of other features shown to improve the solver, described in [Davies and Bacchus, 2013a; 2013b; Saikko *et al.*, 2016a]. We compare the performance of MaxHS with and without reduced cost fixing, with all other features unchanged. We utilized IBM CPLEX v12.7 as the IP/LP solver, and ran our experiments on computing nodes with Xeon 2.8-GHz cores and 256-GB RAM. We limited MaxHS to 1800 seconds and 3.5 GB on each instance. We also report on longer 5-hour (18,000 s), 5-GB runs on Xeon 2.0-GHz cores and 256-GB RAM.

We experimented with all non-random instances that have been collected and made available by the MaxSAT Evaluation during the years 2008–2016. These include extra submitted benchmarks never used in the evaluation. After pruning duplicate instances this yielded 6290 MaxSAT instances (4361 unweighted, 1929 weighted). For the 5-hour runs, however, we omitted 507 unweighted instances with no hard clauses (MS instances) most of which encode MaxCut on random graphs. Core-based solvers, including IHS solvers, perform poorly on such instances, and we did not expect any of these instances to complete in 5 hours with or without reduced cost fixing. This left 5783 instances for these longer experiments (4361 unweighted, 1422 unweighted).

Figure 2 left shows a histogram of the instances grouped by the number of soft clauses that become fixed during solving.

In 5024 of the 6290 instances no reduced cost fixing ever occurs (3953 unweighted, 1071 weighted), but in the remaining 1266 instances fixing can be quite common—in 791 of these instances 100 or more fixings occurred. In extreme cases over a million soft clauses were fixed by the technique.

There were 26 instances where fixing took more than 100 seconds. However, 25 of these were not solvable with or without fixing (22 were MaxCut on random graphs). On one solved instance fixing required 214 seconds out of a total solve time of 835 seconds (this instance was solved in 416 seconds without fixing). Of the remaining 6264 instances, on 1782 instances fixing took zero seconds (LP solving was never invoked since the gap between UB and LB was never small enough), on 3746 instances fixing took less than 1 second, on 298 instances fixing took between 1 and 10 seconds, and on 438 instances fixing took more than between 10 and 100 seconds. Figure 2 middle shows, however, that on these 438 instances fixing is well worth the time it takes. The scatter plot shows that fixing provides a significant speedup for most of these instances, especially on the harder instances.

Figure 2 right shows the performance improvement obtained from reduced cost fixing under a 5-hour per-instance time limit. Here we computed the speedup ratio for each instance, i.e., the CPU time taken without reduced cost fixing divided by the CPU time taken when reduced cost fixing is used. As this ratio will be between 0 and 1 for instances that are slowed down by fixing, we took $\log_2$ of this ratio which produces a symmetry between speedups and slowdowns. The plot is in the form of a histogram showing for how many instances experience various ranges of the $\log$ speedup, and it clearly shows the value of our approach to employing reduced cost fixing in IHS based MaxSAT solvers.

References

[Alviano *et al.*, 2015] Mario Alviano, Carmine Dodaro, and Francesco Ricca. A MaxSAT algorithm using cardinality constraints of bounded size. In *Proc. IJCAI*, pages 2677–2683, 2015.

[Ansótegui *et al.*, 2013] Carlos Ansótegui, Maria Luisa Bonet, Joel Gabàs, and Jordi Levy. Improving WPM2 for (weighted) partial MaxSAT. In *Proc. CP*, volume 8124 of *LNCS*, pages 117–132. Springer, 2013.

[Ansótegui *et al.*, 2015] Carlos Ansótegui, Frédéric Didier, and Joel Gabàs. Exploiting the structure of unsatisfiable cores in MaxSAT. In *Proc. IJCAI*, pages 283–289, 2015.

[Bacchus *et al.*, 2017] Fahiem Bacchus, Antti Hyttinen, Matti Järvisalo, and Paul Saikko. Reduced cost fixing in MaxSAT. In *Proc. CP*, volume 10416 of *LNCS*, pages 641–651. Springer, 2017.

[Chvátal, 1983] Vašek Chvátal. *Linear Programming*. Freeman, 1983.

[Crowder *et al.*, 1983] Harlan Crowder, Ellis L. Johnson, and Manfred Padberg. Solving large-scale zero-one linear programming problems. *Operations Research*, 31(5):803–834, 1983.

[Danzig *et al.*, 1954] George B. Danzig, Delbert R. Fulkerson, and Selmer M. Johnson. Solution of a large-scale traveling-salesman problem. *Operations Research*, 2:393–410, 1954.

[Davies and Bacchus, 2011] Jessica Davies and Fahiem Bacchus. Solving MAXSAT by solving a sequence of simpler SAT instances. In *Proc. CP*, volume 6876 of *LNCS*, pages 225–239. Springer, 2011.

[Davies and Bacchus, 2013a] Jessica Davies and Fahiem Bacchus. Exploiting the power of MIP solvers in MaxSAT. In *Proc. SAT*, volume 7962 of *LNCS*, pages 166–181. Springer, 2013.

[Davies and Bacchus, 2013b] Jessica Davies and Fahiem Bacchus. Postponing optimization to speed up MAXSAT solving. In *Proc. CP*, volume 8124 of *LNCS*, pages 247–262. Springer, 2013.

[Davies, 2013] Jessica Davies. *Solving MAXSAT by Decoupling Optimization and Satisfaction*. PhD thesis, University of Toronto, 2013. http://www.cs.toronto.edu/~jdavies/Davies_Jessica_E_201311_PhD_thesis.pdf.

[Focacci *et al.*, 1999] Filippo Focacci, Andrea Lodi, and Michela Milano. Cost-based domain filtering. In *Proc. CP*, volume 1713 of *LNCS*, pages 189–203. Springer, 1999.

[Heras *et al.*, 2012] Federico Heras, António Morgado, and João Marques-Silva. Lower bounds and upper bounds for MaxSAT. In *Proc. LION 6*, volume 7219 of *LNCS*, pages 402–407. Springer, 2012.

[Karp, 2010] Richard M. Karp. Implicit hitting set problems and multi-genome alignment. In *Proc. CPM*, volume 6129 of *LNCS*, page 151. Springer, 2010.

[Koshimura *et al.*, 2012] Miyuki Koshimura, Tong Zhang, Hiroshi Fujita, and Ryuzo Hasegawa. QMaxSAT: A partial Max-SAT solver. *Journal on Satisfiability, Boolean Modeling and Computation*, 8(1/2):95–100, 2012.

[Li *et al.*, 2006] Chu Min Li, Felip Manyà, and Jordi Planes. Detecting disjoint inconsistent subformulas for computing lower bounds for Max-SAT. In *Proc. AAAI*, pages 86–91. AAAI Press, 2006.

[Li *et al.*, 2008] Chu Min Li, Felip Manyà, Nouredine Ould Mohamedou, and Jordi Planes. Transforming inconsistent subformulas in MaxSAT lower bound computation. In *Proc. CP*, volume 5202 of *LNCS*, pages 582–587. Springer, 2008.

[Lin and Su, 2007] Han Lin and Kaile Su. Exploiting inference rules to compute lower bounds for MAX-SAT solving. In *Proc. IJCAI*, pages 2334–2339, 2007.

[Lin *et al.*, 2008] Han Lin, Kaile Su, and Chu Min Li. Within-problem learning for efficient lower bound computation in Max-SAT solving. In *Proc. AAAI*, pages 351–356. AAAI Press, 2008.

[Martins *et al.*, 2014] Ruben Martins, Vasco M. Manquinho, and Inês Lynce. Open-WBO: A modular MaxSAT solver. In *Proc. SAT*, volume 8561 of *LNCS*, pages 438–445. Springer, 2014.

[Moreno-Centeno and Karp, 2013] Erick Moreno-Centeno and Richard M. Karp. The implicit hitting set approach to solve combinatorial optimization problems with an application to multigenome alignment. *Operations Research*, 61(2):453–468, 2013.

[Morgado *et al.*, 2012] António Morgado, Federico Heras, and João Marques-Silva. Improvements to core-guided binary search for MaxSAT. In *Proc. SAT*, volume 7317 of *LNCS*, pages 284–297. Springer, 2012.

[Morgado *et al.*, 2014] António Morgado, Carmine Dodaro, and Joao Marques-Silva. Core-guided MaxSAT with soft cardinality constraints. In *Proc. CP*, volume 8656 of *LNCS*, pages 564–573. Springer, 2014.

[Nemhauser and Wolsey, 1999] George L. Nemhauser and Laurence A. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience, 1999.

[Saikko *et al.*, 2016a] Paul Saikko, Jeremias Berg, and Matti Järvisalo. LMHS: A SAT-IP hybrid MaxSAT solver. In *Proc. SAT*, volume 9710 of *LNCS*, pages 539–546. Springer, 2016.

[Saikko *et al.*, 2016b] Paul Saikko, Johannes Peter Wallner, and Matti Järvisalo. Implicit hitting set algorithms for reasoning beyond NP. In *Proc. KR*, pages 104–113. AAAI Press, 2016.

[Saikko, 2015] Paul Saikko. Re-implementing and extending a hybrid SAT-IP approach to maximum satisfiability. Master's thesis, University of Helsinki, 2015. http://hdl.handle.net/10138/159186.

[Thorsteinsson and Ottosson, 2002] Erlendur S. Thorsteinsson and Greger Ottosson. Linear relaxations and reduced-cost based propagation of continuous variable subscripts. *Annals of Operations Research*, 115(1-4):15–29, 2002.

[Wolsey, 1998] Laurence A. Wolsey. *Integer Programming*. John Wiley, 1998.

[Yunes *et al.*, 2010] Tallys H. Yunes, Ionut D. Aron, and John N. Hooker. An integrated solver for optimization problems. *Operations Research*, 58(2):342–356, 2010.