An Empirical Study of Encodings for Group MaxSAT [⋆]

Federico Heras, Antonio Morgado, and Joao Marques-Silva

CASL/CSI, University College Dublin, Dublin, Ireland
{fheras,ajrm,jpms}@ucd.ie

Abstract. *Weighted Partial MaxSAT* (WPMS) is a well-known optimization variant of Boolean Satisfiability (SAT) that finds a wide range of practical applications.WPMS divides the formula in two sets of clauses: The *hard* clauses that must be satisfied and the *soft* clauses that can be unsatisfied with a penalty given by their associated *weight*. However, some applications may require each constraint to be modeled as a *set* or *group* of clauses. The resulting formalism is referred to as *Group MaxSAT*. This paper overviews *Group MaxSAT*, and shows how several optimization problems can be modeled as Group MaxSAT. Several *encodings* from Group MaxSAT to standard MaxSAT are formalized and refined. A comprehensive empirical study compares the performance of several MaxSAT solvers with the proposed encodings. The results indicate that, depending on the underlying MaxSAT solver and problem domain, the solver may perform better with a given encoding than with the others.

1 Introduction

Weighted Partial MaxSAT (WPMS) [6] is a well-known optimization variant of Boolean Satisfiability (SAT) that finds a wide range of practical applications [6]. WPMS divides the formula in a set of *hard* clauses that must be satisfied and a set of *soft* clauses that can be unsatisfied with a penalty corresponding to the *weight* associated to the soft clause. Problems modeled as WPMS associate an independent *soft* clause to each original constraint that can be unsatisfied by incurring the associated weight.

However, some applications may require each constraint to be modeled as a *set* or *group* of clauses. In this case, each original constraint is modeled as a *soft group*. As a result, any set of unsatisfied clauses belonging to the same soft group contribute with a *unique weight*. The objective is to find an assignment that satisfies all hard clauses and minimizes the sum of weights of unsatisfied soft groups. This formalism is referred to as *Group MaxSAT* and was introduced in [4] restricted to soft groups with weights 1.

This paper presents the Group MaxSAT problem and shows how several problems can be modeled as Group MaxSAT including soft versions of *CSP*s

[⋆] This work was partially supported by SFI PI grant BEACON (09/IN.1/I2618).

such as *MaxCSP*, *Weighted Boolean Optimization* (*WBO*) and an optimization version of the *Quasi Group Completion* problem. Then, three *encodings* from Group MaxSAT to (*standard*) MaxSAT are presented. The *double weight* encoding is a refinement of the one presented in [4]. The $\top$-encoding is an extension of the double weight encoding. Both the double weight and $\top$-encodings add *additional* variables and clauses. The *straight* encoding essentially encodes the Group MaxSAT formula into a MaxSAT formula *without* adding any additional variables or clauses. But, this paper shows that the straight encoding can only be applied *soundly* whenever the Group MaxSAT formula respects a specific property. The empirical investigation analyzes the performance of *branch and bound* and *core-guided* MaxSAT algorithms [6] with the different encodings. The results indicate that, depending on the underlying MaxSAT solver and problem domain, the solver may perform better with a given encoding than with the others.

Note that recent practical applications have been modeled as WPMS implicitly using a $\top$-like encoding without realizing the general framework that can be associated with them [1, 11]. This endorses the relevance of this study.

The remainder of this paper is organized as follows. Section 2 formally presents the MaxSAT and *Group MaxSAT* problems. Section 3 introduces the three encodings. Section 4 shows how to model several problems into Group MaxSAT. Section 5 presents the empirical study that analyzes the *performance* of several MaxSAT solvers with the different encodings. Finally, section 6 overviews the related work and section 7 concludes the paper.

2 The Group MaxSAT problem

This section introduces the necessary definitions and notation related to the MaxSAT and *group* MaxSAT problems.

MaxSAT Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of Boolean variables. A *literal* is either a variable x_i or its negation $\bar{x}_i$. The variable to which a literal l refers is denoted by $var(l)$. Given a literal l, its negation $\bar{l}$ is $\bar{x}_i$ if l is x_i and it is x_i if l is $\bar{x}_i$. A *clause* C is a disjunction of literals. An *assignment* is a set of literals $\mathcal{A} = \{l_1, l_2, \ldots, l_n\}$ such that for all $l_i \in \mathcal{A}$, its variable $var(l_i) = x_i$ is assigned to $true$ or $false$. If variable x_i ($\bar{x}_i$) is assigned to $true$, literal x_i ($\bar{x}_i$) is *satisfied* and literal $\bar{x}_i$ (x_i) is *unsatisfied*. If all variables in X are assigned, the assignment is called *complete*. An assignment *satisfies* a literal iff it belongs to the assignment, it satisfies a clause iff it satisfies one or more of its literals and it *unsatisfies* a clause iff it contains the negation of all its literals.

A *weighted* clause is a pair (C_i, w_i), where C_i is a clause and w_i is the cost of unsatisfying it, also called its *weight*. Many real problems contain *mandatory* (or *hard*) clauses that *must* be satisfied which are associated with a special weight $\top$. Non-mandatory clauses are also called *soft* clauses. A MaxSAT formula is $\varphi = \varphi_H \cup \varphi_S$ where φ_H is a set of *hard* clauses, and φ_S is a set of *soft* clauses. A *model* is a complete assignment $\mathcal{A}$ that satisfies all mandatory clauses. The *cost of a model* is the sum of weights of the soft clauses that it unsatisfies.

Group MaxSAT A group MaxSAT formula is $\psi = \psi_H \cup G_S$ where ψ_H is a set of hard clauses and $G_S = \{(G_1, w_1), \ldots, (G_m, w_m)\}$ is a set of *soft groups*. Each group $(G_i, w_i) \in G_S$ is defined by a set of clauses $G_i = \{C_{i1}, \ldots, C_{ik}\}$ and a weight w_i. Any assignment that unsatisfies a subset of the clauses in a soft group (G_i, w_i) will be penalized with a unique cost of w_i. The objective of the *group MaxSAT* problem is to find an assignment that satisfies all hard clauses and minimizes the sum of weights of unsatisfied soft groups. Note that when each soft group is formed by just one clause, the group MaxSAT formula actually represents a standard (partial) MaxSAT problem.

3 Encoding Group MaxSAT as standard MaxSAT

This section develops several encodings from Group MaxSAT to MaxSAT which allow to solve Group MaxSAT with state-of-the-art MaxSAT solvers. In the remaining of this section, ψ refers to a Group MaxSAT formula and φ refers to MaxSAT formula.

Definition 1 (Double weight encoding). *Each hard clause $(C, \top) \in \psi$ becomes a hard clause in φ. Let $G_i = \{C_{i1}, \ldots, C_{ik}\}$ be the set of clauses in a soft group $(G_i, w_i) \in \psi$. Each clause in G_i is extended with the same additional variable r and a weight $2w_i$ is given to the resulting soft clause which is added to φ. Additionally, the unit soft clause $(\bar{r}, w_i)$ is also added to φ. Hence, the soft group (G_i, w_i) is translated into the soft clauses $\{(C_{i1} \vee r, 2w_i), \ldots, (C_{ik} \vee r, 2w_i), (\bar{r}, w_i)\}$.*

Observe that the original double weight encoding introduced in [4] represented hard clauses as soft weighted clauses, but without actually *declaring* them as hard clauses using $\top$. Nevertheless, current MaxSAT solvers can take advantage if the clauses are *explicitly declared* as hard. For this reason, the double weight encoding has been reformulated in this way. Moreover, the clauses with double weight can also be made hard without changing the semantics of the encoding. This results in the $\top$-*encoding*.

Definition 2 ($\top$-encoding). *The hard clauses in ψ are simply added to φ as hard clauses. Let $G_i = \{C_{i1}, \ldots, C_{ik}\}$ be the set of clauses in a soft group $(G_i, w_i) \in \psi$. Each clause in G_i is extended with the same additional variable r and a weight $\top$ is given to the resulting hard clause which is added to φ. Additionally, the unit soft clause $(\bar{r}, w_i)$ is also added to φ. Hence, the soft group (G_i, w_i) is translated to the clauses $\{(C_{i1} \vee r, \top), \ldots, (C_{ik} \vee r, \top), (\bar{r}, w_i)\}$, where all clauses are hard except the unit clause $(\bar{r}, w_i)$.*

One more encoding is considered in this paper which is referred to as *straight encoding*. Essentially, the Group MaxSAT formula is encoded into a MaxSAT formula *without* adding additional variables or clauses.

Definition 3 (Straight encoding). *Each hard clause $(C, \top) \in \psi$ becomes a hard clause in φ. For each clause C of a soft group $(G_i, w_i) \in \psi$ (i.e. $C \in G_i$), the soft clause (C, w_i) is added to φ.*

However, such encoding may not maintain the semantics of the original Group MaxSAT formula (i.e. any assignment in the original Group MaxSAT formula should have exactly the same cost in the resulting MaxSAT formula). To determine whether the straight encoding can be applied or not, the following property has been identified.

Property 1 (One unsat property). The Group MaxSAT formula ψ is said to satisfy the *one unsat property* (*OUP*) if any optimal model $\mathcal{A}$ unsatisfies *at most* one clause for each soft group of ψ.

Remark 1. Given a Group MaxSAT formula ψ that satisfies the one unsat property (i.e. OUP(ψ) is true), the straight encoding can be safely applied to ψ.

4 Modeling problems as Group MaxSAT

This section introduces several optimization problems and shows how to model them as a Group MaxSAT formula. Then, it is studied which encodings into MaxSAT can be applied to the proposed modeling. In particular, the double weight and $\top$-encodings can always be applied. Differently, the straight encoding can only be applied if the modeling respects the one unsat property.

Weighted Boolean Optimization. A *Weighted Boolean Optimization* instance (*WBO*) [15] is composed by a set of *soft constraints PB_s* and a set of *hard constraints PB_h*. Each soft constraint $(PB_i, w_i) \in PB_s$ has associated a weight $w_i > 0$. Each constraint in WBO is a *pseudo-Boolean* (*PB*) constraint that can be translated to clauses using any of the available encodings in the literature [7]. A PB constraint PB_i has the form $\sum_{j=1}^{n} a_{ij} l_j \geq b_i$, where $x_j \in \{0, 1\}$, l_j is either x_j or $1 - x_j$, and c_j, a_{ij} and b_i are non-negative integers. The WBO problem consists in finding an assignment that satisfies all hard constraints such that the sum of weights of unsatisfied soft constraints is minimized. A WBO problem can be casted into Group MaxSAT: (1) each hard constraint $PB_i \in PB_h$ is translated to a set of hard clauses [7], (2) each soft constraint $(PB_i, w_i) \in PB_s$ is translated to a set of clauses [7] that becomes a soft group (G_i, w_i).

Proposition 1. *The one unsat property does not hold in general for a Group MaxSAT formula ψ built as stated above (i.e. OUP(ψ) is false).*

Proof. Consider a WBO problem with four soft PB constraints: $(\bar{x}_1 + \bar{x}_2 + \bar{x}_3 \geq 1, 2), (x_1 \geq 1, 1), (x_2 \geq 1, 1), (x_3 \geq 1, 1)$. A possible translation to clauses and to soft groups of the PB constraints (using the *pairwise encoding*) is: $(G_1, 1) = \{(\bar{x}_1 \vee \bar{x}_2), (\bar{x}_1 \vee \bar{x}_3), (\bar{x}_2 \vee \bar{x}_3)\}$, $(G_2, 1) = \{(x_1)\}$, $(G_3, 1) = \{(x_2)\}$ and $(G_4, 1) = \{(x_3)\}$. An optimal assignment is $\mathcal{A} = \{x_1, x_2, x_3\}$ (with cost of 1) where just the first PB constraint is unsatisfied but the *three* clauses representing it are unsatisfied *simultaneously* in the same group G_1.

Binary MaxCSP. The *MaxCSP* problem is a well-known optimization version of the CSP problem. A MaxCSP is defined by a set of variables $X = \{X_1, \ldots, X_n\}$. Each variable X_i has a domain of values $D_i = \{1, 2, \ldots, d_i\}$ that can take such variable. Several constraints $C = \{C_1, \ldots, C_m\}$ are defined over subsets of variables. This paper considers *binary* MaxCSP where constraints only involve pairs of variables. This means that having n variables, there can be at most $(n \times (n-1))/2$ constraints. Each binary constraint C_j defined over variables X_{j1} and X_{j2} with respective domains D_{j1} and D_{j2} is formed by a set of tuples $C_j = \{t_1, \ldots, t_k\}$. Each tuple $t_k \in C_j$ forbids a *simultaneous* assignment of the two variables in $|D_{j1} \times D_{j1}|$ (Cartesian product). There are several ways to encode a CSP problem into clauses [16]. This paper just considers the *direct encoding* but any other could be considered.

The direct encoding associates a Boolean variable x_{ij} with each value j of a MaxCSP variable X_i with domain of d values and $1 \leq j \leq d$. For each MaxCSP variable X_i with values $\{1, 2, \ldots, d\}$ the following hard clauses are added to the formula $(x_{i1} \vee x_{i2} \vee \cdots \vee x_{id}, \top)$ (1), to ensure that each MaxCSP variable is given a value. For each MaxCSP variable X_i and each pair of different values j, k of X_i, the hard clause $(\bar{x}_{ij} \vee \bar{x}_{ik}, \top)$ is added, to ensure that the variable X_i is not given more than one value (2). Finally, for each tuple t_k belonging to the same constraint C_j (i.e. $t_k \in C_j$) a binary clause representing its contribution is added to the same soft group $(G_j, 1)$.

Proposition 2. *A Group MaxSAT formula ψ built as stated above always respects the one unsat property (i.e. OUP(ψ) is true).*

Proof. Consider an optimal model $\mathcal{A}$ for ψ. Without loss of generality, let $(G_j, 1)$ be the soft group defined for the constraint C_j involving the MaxCSP variables X_1 and X_2 both with k values. Since $\mathcal{A}$ is optimal, then all hard clauses are satisfied, in particular (2). Therefore at most one of the Boolean variables in $x_{11}, \ldots, x_{1k}$ is assigned to true and at most one of the Boolean variables in $x_{21}, \ldots, x_{2k}$ is assigned to true. Thus, *at most one tuple* is unsatisfied in C_j and consequently in the soft group $(G_j, 1)$.

Note that the straight encoding of the proposed Group MaxSAT formula results in a MaxSAT formula which is identical to applying the *direct encoding* from MaxCSP to MaxSAT in [3, 8]. Additionally, note that the *logarithmic* [8] and *minimal support encoding* [3] produce a Group MaxSAT formula that respects the one unsat property, whereas the *support encoding* [3] does not.

Quasigroup Completion Problem. Given an $n \times n$ matrix and n colors, a *latin square* of order n is a colored matrix such that all cells are colored, each color occurs exactly once in each row and each color occurs exactly once in each column. The *Quasigroup completion problem* [12] (*QCP*) is a latin square where a percentage of the cells have been initially colored. The task of interest is to decide whether the partial quasigroup (latin square) can be completed so that a full quasigroup is obtained. The *minimum encoding* [12] of QCP into a SAT problem requires:

1. n^3 Boolean variables. Variable x_{ijk} represents cell i, j with color k where $i, j, k = \{1, 2, \dots, n\}$.
2. Some color to be assigned to each cell i, j: $(x_{ij1} \vee x_{ij2} \vee \dots \vee x_{ijn})$.
3. No color to be repeated for row i: $\{(\bar{x}_{i1k} \vee \bar{x}_{i2k}) \wedge (\bar{x}_{i1k} \vee \bar{x}_{i3k}) \wedge \dots (\bar{x}_{i1k} \vee \bar{x}_{ink})\}$ for $k = \{1, 2, \dots, n\}$. Hence, a total of n *row constraints.*
4. No color to be repeated for column j: $\{(\bar{x}_{1jk} \vee \bar{x}_{2jk}) \wedge (\bar{x}_{1jk} \vee \bar{x}_{3jk}) \dots (\bar{x}_{1jk} \vee \bar{x}_{njk})\}$ for $k = \{1, 2, \dots, n\}$. Hence, a total of n *column constraints.*
5. For each initially colored cell i, j with color k, the unit clause x_{ijk} is added.

This paper considers the optimization version of the QCP proposed in [4]. In this case, the goal is to find an assignment that minimizes the total number of unsatisfied *row* and *column* constraints. To model such optimization problem in group MaxSAT the set of variables is n^3 as stated in 1. (above). Each clause in 2. becomes a hard clause. All the clauses associated to each row constraint i 3. become a soft group $(G_i, 1)$. All clauses associated to each column constraint j 4. constitute a soft group $(G_j, 1)$. Finally, each unit clause that represents a colored cell in 5. is added as a hard clause.

Proposition 3. *The one unsat property does not hold in general for a Group MaxSAT formula ψ built as stated above (i.e. OUP(ψ) is false).*

Proof.

$$A = \begin{bmatrix} 1 & x & x \\ x & 2 & 3 \\ x & 3 & 2 \end{bmatrix} \qquad A^S = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 2 \end{bmatrix}$$

Let A be the matrix of order 3 above with some initial assignments. Cells with value x means the color is undecided. A possible optimal solution is A_S. Such solution unsatisfies exactly one row constraint and one column constraint, and at the same time unsatisfies more than one clause in the group representing the unsatisfied row constraint and unsatisfies more than one clause in the the group representing the unsatisfied column constraint.

Crafted Group MaxSAT instances. Let $\langle v, k, r \rangle$ be three natural numbers. A set of k clauses are randomly created of size 3 involving variables in $x_1, \dots, x_v$. Then a SAT solver is called to obtain a satisfying model $\mathcal{A} = \{l_1, l_2, \dots, l_v\}$. If no such model exists, then the k clauses are generated again. This process is repeated until a set of k clauses with a model is found. Then, one of the literals is randomly selected and *flipped* in the assignment $\mathcal{A}$. Given a literal l_i in $\mathcal{A}$, if l_i is x_i then a unit clause x_i is created, otherwise $\bar{x}_i$ is created. All the unit clauses form a soft group with weight 1 (i.e. $(G_i, 1)$). Additionally, hard clauses are added: (1) the set of k clauses; (2) the clause $(l_1 \vee l_2 \vee \dots \vee l_v)$; and (3) the constraint $\sum_{i=1}^{v} x_i \leq 1$ translated to hard clauses. This process is repeated r times resulting in a Group MaxSAT instance with r soft groups.

Proposition 4. *A Group MaxSAT formula ψ built as stated above always respects the one unsat property (i.e. OUP(ψ) is true).*

Solver	Partial		Soft		Total
	Sol	Time	Sol	Time	
$\#I$	536	-	201	-	737
BC-T	431	84.11	157	64.92	588
BC-D	320	111.06	28	127.13	348
BCD-T	432	106.77	157	54.99	589
BCD-D	322	116.46	28	150.97	350
PM1-T	452	109.82	165	79.36	617
PM1-D	442	158.29	153	97.21	595
MM-T	366	70.02	37	228.23	403
MM-D	93	101.7	2	1.64	95
SAT4J	349	97.33	161	107.26	510

Table 1. Results on WBO instances.

Proof. Consider an optimal model $\mathcal{A}$ for ψ. By contradiction, suppose that $\mathcal{A}$ unsatisfies more than one clause in a soft group G_i. This means that such assignment $\mathcal{A}$ also unsatisfies the constraint (3). Since (3) are hard clauses, then $\mathcal{A}$ cannot be a model for ψ.

5 Experimental Evaluation

The empirical evaluation studies the effect on the performance of complete MaxSAT solvers on each of the encodings in Group MaxSAT formulas. Experiments were conducted on a HPC cluster with 50 nodes, each node is a CPU Xeon E5450 3GHz, 32GB RAM and Linux. For each run, the time limit was set to 1200 seconds and a memory limit of 4GB.

Current most effective *complete* MaxSAT solvers are either based on *branch and bound* or on iteratively calling a SAT oracle. MiniMaxSat (*MM*) [9] was selected among other branch and bound solvers, because it takes additional advantage of *clause learning* and *backjumping* on the instances considered. 3 solvers based on calling a SAT oracle were considered. The first one is WPM1 (*PM1*) [2]. PM1 is characterized by refining a *lower bound*, adding *AtMost1* constraints to the formula and adding more than one *relaxation variable* per soft clause. The second one is *core-guided binary search* (*BC*) [10]. BC is characterized by refining both a *lower bound* and *upper bound*, adding *AtMostK* constraints to the formula and adding *at most one* relaxation variable per soft clause. The third one is *core-guided binary search with disjoint cores* (*BCD*) [10] which is an extended version of BC that additionally exploits the information of *disjoint cores*. Given a MaxSAT solver X, it can be executed to solve a Group MaxSAT instance using the straight encoding S, the double weight encoding D or the $\top$-encoding. This is noted as X-S, X-D and X-T, respectively.

The results for the WBO instances are presented in Table 1 and the WBO dedicated solver SAT4J [5] is added to the comparison. All WBO instances with *small integers* (*smallint*) track of the *Pseudo-Boolean Competition 2011* were considered. Such instances are divided into two categories: *partial* and *soft smallint*. The first column shows the name of the solver. The second, fourth and sixth columns, show the total number of solved instances by each solver for the partial

n	$\%p$	$\#I$	BC-T		BC-D		BCD-T		BCD-D		PM1-T		PM1-D		MM-T		MM-D	
			Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time
16	45	10	9	0.44	9	0.82	9	0.41	9	0.71	9	0.22	9	0.31	7	0.48	0	0.00
18	45	10	10	9.21	10	14.90	10	9.25	10	9.32	10	10.20	10	20.28	5	0.14	0	0.00
20	45	10	10	1.38	10	1.86	10	0.99	10	1.19	10	0.47	10	0.70	2	0.23	0	0.00
22	45	10	10	2.25	10	2.54	10	1.41	10	1.97	10	0.97	10	1.02	3	0.34	0	0.00
16	60	10	10	0.71	10	3.80	10	0.89	10	6.19	10	0.36	10	1.13	10	2.28	0	0.00
18	60	10	10	1.42	10	8.24	10	1.75	10	3.45	10	0.61	10	1.14	10	14.15	0	0.00
20	60	10	10	2.46	10	13.86	10	1.82	10	23.14	10	0.94	10	2.49	10	64.46	0	0.00
22	60	10	10	4.96	9	12.09	10	2.95	8	57.60	10	8.53	10	31.09	8	194.18	0	0.00
16	75	10	10	0.95	10	40.67	10	1.23	10	6.47	10	0.39	10	0.84	10	21.04	1	48.34
18	75	10	10	2.10	10	322.60	10	3.44	10	19.96	10	0.69	10	1.13	10	36.81	1	963.19
20	75	10	10	3.19	5	550.65	10	3.46	10	7.93	10	1.01	10	2.55	9	253.15	0	0.00
22	75	10	10	5.73	2	588.16	10	5.85	8	20.61	10	1.94	10	3.02	3	209.55	0	0.00
-	-	120	119	-	105	-	119	-	115	-	119	-	119	-	87	-	2	-

Table 2. Results on QCP instances (BC, BCD, PM1 and MM).

category, the soft category and the sum of both, respectively. The third and fifth columns display the average time of the solved instances by each solver for the partial and soft categories, respectively. Observe that in this set of instances all MaxSAT solvers obtain their best performance with the $\top$-encoding. The double weight encoding is worst performing option, specially for BC, BCD and MM. The solvers based on calling a SAT oracle (PM1, BC and BCD) perform quite well on these instances, even better than the dedicated solver (SAT4J) within the specified memory and time limits.

The remaining tables present results for QCP, crafted Group MaxSAT and MaxCSP. Each table shows in the left most columns the parameters used to create each set of instances, and for each set, 10 instances were created ($\#I$). The remaining pairs of columns show the number of solved instances (Sol) and the average time (Time) of the solved instances for each solver.

Table 2 presents the results for QCP instances. For such instances, the order n of the latin square is fixed and the percentage of randomly preassigned cells $\%p$ is varied from 45% to 75% (step 15%). The best solver for this benchmark suite is PM1. All MaxSAT solvers perform worse with the double weight encoding and such worsening is specially evident in MM being orders of magnitude worse than with the $\top$-encoding. Note that when using the double weight encoding, maintaining disjoint cores (BCD) improves several orders of magnitude the performance with respect the non-disjoint version (BC).

Tables 3 and 4 present the results for the crafted Group MaxSAT instances defined by values $\langle v, k, r \rangle$ where v and k are the number of variables and the number of clauses per repetition, respectively, and r is the the total number of repetitions. The double weight encoding, depending on the solver, is the worst option or the second option but never the best option. PM1 performs better with the straight encoding, whereas BC and MM prefer the $\top$-encoding. The differences in performance are quite small for all the three encodings for BCD in this particular problem, the straight and $\top$ being slightly better than the double weight encoding. Note that both BC and BCD apply a *lower bound heuristic* [10]

v	k	r	$\#I$	BC-S		BC-T		BC-D		BCD-S		BCD-T		BCD-D	
				Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time
10	20	50	10	10	0.09	10	0.10	10	12.40	10	0.07	10	0.07	10	0.10
10	20	75	10	10	0.28	10	0.19	9	44.75	10	0.14	10	0.15	10	0.22
10	20	100	10	10	0.48	10	0.29	6	5.13	10	0.25	10	0.27	10	0.36
50	100	100	10	10	9.48	10	9.72	3	417.53	10	7.67	10	8.55	10	12.31
50	100	200	10	10	50.35	10	45.60	0	0.00	10	32.23	10	34.85	10	48.04
50	100	300	10	10	103.32	10	114.59	0	0.00	10	97.81	10	98.84	10	110.77
50	100	400	10	10	182.59	10	230.25	0	0.00	10	172.07	10	167.02	10	201.52
50	100	500	10	10	311.43	10	289.33	0	0.00	10	226.40	10	352.32	10	349.13
50	100	600	10	10	357.53	10	369.83	0	0.00	10	294.56	10	353.49	10	390.09
50	100	700	10	10	538.78	10	442.91	0	0.00	10	446.19	10	525.32	10	591.83
50	100	800	10	10	650.02	10	595.21	0	0.00	10	534.42	10	586.20	10	679.10
50	100	900	10	6	797.94	10	708.68	0	0.00	10	669.56	10	744.14	10	852.80
50	100	1000	10	9	926.82	10	875.04	0	0.00	10	849.50	10	889.14	8	1080.78
-	-	-	130	125	-	130	-	28	-	130	-	130	-	128	-

Table 3. Crafted Group MaxSAT instances (BC and BCD).

v	k	r	$\#I$	PM1-S		PM1-T		PM1-D		MM-S		MM-T		MM-D	
				Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time
10	20	50	10	10	0.05	10	0.05	10	0.08	10	0.74	10	0.42	10	0.61
10	20	75	10	10	0.10	10	0.10	10	0.16	10	3.60	10	1.54	10	2.12
10	20	100	10	10	0.17	10	0.17	10	0.28	10	11.21	10	3.88	10	6.16
50	100	100	10	10	2.37	10	2.75	10	5.00	10	789.31	10	47.14	10	160.01
50	200	100	10	10	8.42	10	11.38	10	20.80	0	0.00	10	433.20	9	759.81
50	300	100	10	10	19.07	10	25.67	10	43.00	0	0.00	0	0.00	0	0.00
50	400	100	10	10	33.54	10	43.40	10	66.93	0	0.00	0	0.00	0	0.00
50	500	100	10	10	58.26	10	74.10	10	121.91	0	0.00	0	0.00	0	0.00
50	600	100	10	10	82.90	10	94.54	10	173.30	0	0.00	0	0.00	0	0.00
50	700	100	10	10	103.78	10	154.66	10	211.66	0	0.00	0	0.00	0	0.00
50	800	100	10	10	137.21	10	209.36	10	266.36	0	0.00	0	0.00	0	0.00
50	900	100	10	10	182.78	10	240.67	10	388.27	0	0.00	0	0.00	0	0.00
50	1000	100	10	10	230.87	10	312.95	10	478.02	0	0.00	0	0.00	0	0.00
-	-	-	130	130	-	130	-	130	-	40	-	50	-	49	-

Table 4. Crafted Group MaxSAT instances (PM1 and MM).

by default. Additional (but omitted) experiments show that if such heuristic is deactivated, then the $\top$-encoding is orders of magnitude better for BC and BCD.

Tables 5 and 6 present the results for the binary MaxCSP instances. A MaxCSP instance is defined by the values $\langle n, d, \%c, \%t \rangle$ following the protocol introduced in [13]. n is the number of variables and d is the number of values for the domain of each variable. The percentage of constraints $\%c$ determines how many binary constraints the problem has. 100% means that there exists exactly one constraint for each pair of different variables. Finally, $\%t$ determines the *tightness* of each constraint, where 100% means that there are d^2 tuples per constraint. MM is the best MaxSAT solver for this set of benchmarks. BC, BCD and MM were executed in the same set of instances, whereas PM1 was executed on smaller instances given its poor performance. For this benchmark, the double weight encoding is by far the worst encoding for all the solvers (up to several orders of magnitude). BC and BCD prefer the $\top$-encoding over the straight encoding showing improvements of one or two orders of magnitude. In

n	d	$\%c$	$\%t$	$\#I$	BC-S		BC-T		BC-D		BCD-S		BCD-T		BCD-D	
					Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time
15	5	50	65	10	10	19.95	10	1.26	8	615.71	10	35.94	10	0.73	6	542.23
15	5	50	70	10	10	52.98	10	1.69	1	354.56	10	111.67	10	1.10	0	0.00
15	5	50	75	10	10	118.24	10	3.08	0	0.00	10	170.19	10	1.79	0	0.00
15	5	50	80	10	8	342.14	10	2.49	0	0.00	10	308.38	10	2.16	0	0.00
15	5	50	85	10	6	324.32	10	2.09	0	0.00	8	636.09	10	2.07	0	0.00
15	5	50	90	10	6	540.56	10	1.66	0	0.00	6	610.48	10	1.39	0	0.00
12	5	100	65	10	3	489.56	10	25.00	0	0.00	6	720.41	10	27.05	0	0.00
12	5	100	70	10	2	1114.49	10	19.79	0	0.00	4	826.97	10	14.89	0	0.00
12	5	100	75	10	1	837.43	10	18.39	0	0.00	2	867.30	10	19.94	0	0.00
12	5	100	80	10	0	0.00	10	29.14	0	0.00	0	0.00	10	25.02	0	0.00
12	5	100	85	10	0	0.00	10	30.39	0	0.00	0	0.00	10	31.78	0	0.00
12	5	100	90	10	0	0.00	10	9.80	0	0.00	0	0.00	10	11.14	0	0.00
-	-	-	-	120	56	-	120	-	9	-	66	-	120	-	6	-

Table 5. Results on MaxCSP instances (BC and BCD).

n	d	$\%c$	$\%t$	$\#I$	MM-S		MM-T		MM-D	
					Sol	Time	Sol	Time	Sol	Time
15	5	50	65	10	10	0.12	10	2.90	10	1.71
15	5	50	70	10	10	0.18	10	4.67	10	2.64
15	5	50	75	10	10	0.20	10	5.01	10	5.14
15	5	50	80	10	10	0.32	10	6.64	10	11.90
15	5	50	85	10	10	0.39	10	6.13	10	17.65
15	5	50	90	10	10	0.39	10	5.70	10	34.69
12	5	100	65	10	10	0.76	10	7.91	10	14.31
12	5	100	70	10	10	0.68	10	10.38	10	21.02
12	5	100	75	10	10	0.81	10	9.55	10	30.64
12	5	100	80	10	10	1.05	10	11.73	10	63.02
12	5	100	85	10	10	1.51	10	9.48	10	93.29
12	5	100	90	10	10	0.66	10	7.18	10	118.18
-	-	-	-	120	120	-	120	-	120	-

n	d	$\%c$	$\%t$	$\#I$	PM1-S		PM1-T		PM1-D	
					Sol	Time	Sol	Time	Sol	Time
12	5	50	65	10	10	0.37	10	0.81	10	24.07
12	5	50	70	10	10	0.82	10	2.52	9	56.18
12	5	50	75	10	10	8.67	10	19.24	5	175.08
12	5	50	80	10	10	90.64	8	69.15	5	215.80
12	5	50	85	10	8	5.77	6	143.99	3	89.85
12	5	50	90	10	10	120.43	10	14.29	6	364.38
8	5	100	65	10	10	0.66	10	2.81	9	215.19
8	5	100	70	10	10	1.72	10	93.87	4	385.83
8	5	100	75	10	10	8.60	10	195.08	2	180.11
8	5	100	80	10	10	18.65	6	119.94	2	236.33
8	5	100	85	10	10	9.58	10	92.34	1	746.05
8	5	100	90	10	10	2.92	7	1.48	1	113.63
-	-	-	-	120	118	-	107	-	57	-

Table 6. Results on MaxCSP instances (MM and PM1).

contrast, MM and PM1 prefer the straight encoding rather than the $\top$-encoding also showing improvements of one or two orders of magnitude. Finally, note that a dedicated *WCSP* solver [13] (Toulbar2.08) can solve the instances in negligible time (not shown in tables).

Based on these experiments, several conclusions can be drawn. In general, the $\top$-encoding provides substantially better performance than the double weight encoding for any kind of MaxSAT solver. MaxSAT solvers based on calling a SAT oracle and adding *AtMostK* constraints (BC and BCD) prefer the $\top$-encoding to the straight encoding (whenever available). Solvers based on calling a SAT oracle and adding *AtMost1* constraints (PM1), prefer the straight encoding. Finally, branch and bound solvers (MM) may prefer the straight or the $\top$-encoding depending on the specific problem being solved.

6 Related Work

The Group MaxSAT framework was introduced in [4] restricted to soft groups with weights 1 under the name *softCNF*. In [4] the concept of *hard groups* is also considered (i.e. sets of clauses such that any assignment must satisfy all of them) and native branch and bound solver was proposed which is not publicly available. Whenever a clause in a hard group becomes unit after some assignments, *unit propagation* is safely applied. In fact, this is a property of the *partial MaxSAT* problem and it can be applied directly to independent hard clauses of a MaxSAT problem. For this reason, this paper does not consider hard groups. *Early* MaxSAT solvers, predecessors to modern *branch and bound* [9] and *core-guided* MaxSAT solvers [2], are compared in [4] against the native solver by translating the Group MaxSAT problem into MaxSAT using a primitive version of the double weight encoding. Nevertheless, experiments on similar benchmarks (MaxCSP and QCP) indicate that modern MaxSAT solvers with the appropriate encoding are orders of magnitude faster than the native solver in [4].

The one unsat property allows to represent several problems with the *straight encoding*. As noted in the experiments, the straight encoding in the MaxCSP and crafted Group MaxSAT benchmarks is the most appropriate encoding for some MaxSAT solvers. This endorses the relevance of checking the one unsat property introduced in this paper.

In [15] a translation from *WBO* to *Pseudo-Boolean Optimization* (*PBO*) is introduced. A PBO [7] has the form:

minimize $\sum_{j=1}^{n} c_j \cdot x_j$

subject to $\sum_{j=1}^{n} a_{ij} l_j \geq b_i, \quad i = 1 \ldots m$

where $x_j \in \{0,1\}$, l_j is either x_j or $1 - x_j$, and c_j, a_{ij} and b_i are non-negative integers. Each soft PB constraint $(\sum_{j=1}^{n} a_{ij} l_j \geq b_i, w_i)$ is extended with an additional variable r resulting in the PB constraint $\sum_{j=1}^{n} a_{ij} l_j + b_i r \geq b_i$ and the element $\bar{r} \cdot w_i$ is added to the minimization function of a PBO instance. Such encoding can be understood as the PBO counterpart of the $\top$-encoding for MaxSAT. It will be referred to as $\top$-*pbo*-encoding. Hence, WBO can be translated to MaxSAT using the $\top$-*pbo*-encoding to translate the WBO problem into PBO and then from PBO to standard MaxSAT as suggested in [9]. Additional experiments do not show major differences between using the $\top$-pbo-encoding (from WBO to PBO and then from PBO to MaxSAT) and the $\top$-encoding (from Group MaxSAT to MaxSAT) on the current available WBO instances.

In recent years, many practical applications have been modeled into PBO and to MaxSAT implicitly using the $\top$-pbo-encoding and the $\top$-encoding, respectively. Such works focus on the *application itself* rather than in the *framework* used to model and solve the problem. Examples of such applications include the optimization of *area and delay in multiple constant multiplications* [1] (*PBO*) and the *localization of errors in programs* [11] (MaxSAT).

7 Conclusions and Future Work

This paper overviews Group MaxSAT and shows how it can be used to model several optimization problems, including MaxCSP, WBO and an optimization version of QCP. Then, three original encodings from Group MaxSAT to MaxSAT are proposed. The *double weight encoding* is a refinement of the one introduced in [4]. The $\top-$*encoding* extends the later one by declaring some of the clauses as hard. Both the double and $\top$-encodings require additional variables and clauses.

This paper also introduces the *one unsat property* (*OUP*) which allows to characterize which Group MaxSAT formulas can be translated to MaxSAT without additional variables or clauses through the *straight encoding*.

To the best of our knowledge, this paper is the first to address the question of how to encode Group MaxSAT as MaxSAT, and analyzes the impact of the encodings on the performance of MaxSAT solvers. The empirical investigation shed light on practical relevant questions on the best encoding for each MaxSAT solver. This will help to choose the right approach in future applications.

Future research directions include to extend the study with additional Group MaxSAT benchmarks and develop a Group MaxSAT solver based on a *portfolio* of MaxSAT solvers. In particular, one line of work is to create a *competence map* [14] to allow a dedicated Group MaxSAT solver to *automatically* decide which is the most appropriate MaxSAT solver and encoding for a given Group MaxSAT formula.

References

1. L. Aksoy, E. Costa, P. F. Flores, and J. Monteiro. Exact and approximate algorithms for the optimization of area and delay in multiple constant multiplications. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 27(6), 2008.
2. C. Ansótegui, M. L. Bonet, and J. Levy. Solving (weighted) partial MaxSAT through satisfiability testing. In *SAT*, pages 427–440, 2009.
3. J. Argelich, A. Cabiscol, I. Lynce, and F. Manyà. Modelling Max-CSP as partial Max-SAT. In *SAT*, pages 1–14, 2008.
4. J. Argelich and F. Manyà. Exact Max-SAT solvers for over-constrained problems. *Journal of Heuristics*, 12(4-5):375–392, 2006.
5. D. Le Berre and A. Parrain. The Sat4j library, release 2.2. *JSAT*, 7:59–64, 2010.
6. A. Biere, M. Heule, H. Maaren, and T. Walsh, editors. *Handbook of Satisfiability*, 2009.
7. N. Een and N. Sörensson. Translating Pseudo-Boolean Constraints into SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, 2:1–26, 2006.
8. F. Heras, J. Larrosa, S. Givry, and T. Schiex. 2006 and 2007 Max-SAT evaluations: Contributed instances. *Journal on Satisfiability, Boolean Modeling and Computation*, 4(2-4):239–250, June 2008.
9. F. Heras, J. Larrosa, and A. Oliveras. MiniMaxSat: An efficient weighted Max-SAT solver. *Journal of Artificial Intelligence Research*, 31:1–32, January 2008.
10. F. Heras, A. Morgado, and J. Marques-Silva. Core-guided binary search for maximum satisfiability. In *AAAI Conference on Artificial Intelligence*. AAAI, 2011.
11. M. Jose and R. Majumdar. Cause clue clauses: error localization using maximum satisfiability. In *ACM Conference on PLDI*, pages 437–446, 2011.
12. H. A. Kautz, Y. Ruan, D. Achlioptas, C. P. Gomes, B. Selman, and M. E. Stickel. Balance and filtering in structured satisfiable problems. In *IJCAI*, 2001.
13. J. Larrosa and T. Schiex. Solving weighted CSP by maintaining arc consistency. *Artif. Intell.*, 159(1-2):1–26, 2004.
14. Jérôme Maloberti and Michèle Sebag. Fast theta-subsumption with constraint satisfaction algorithms. *Machine Learning*, 55(2):137–174, 2004.
15. V. M. Manquinho, R. Martins, and I. Lynce. Improving unsatisfiability-based algorithms for Boolean optimization. In *SAT*, pages 181–193, 2010.
16. T. Walsh. SAT v CSP. In *CP*, pages 441–456, 2000.