Engineering and Evaluating Multi-objective Pseudo-Boolean Optimizers*

Christoph Jabs, Jeremias Berg, and Matti Järvisalo

Department of Computer Science, University of Helsinki, Helsinki, Finland
{christoph.jabs, jeremias.berg, matti.jarvisalo}@helsinki.fi

Abstract. Various real-world settings give rise to combinatorial optimization problems with multiple conflicting objectives, motivating the development of practical approaches to the challenging task of finding Pareto-optimal solutions to declarative models of multi-objective problems. In this work we focus on multi-objective optimization over pseudo-Boolean constraints (MO-PBO) as an extension of propositional clauses and, at the same time, an important class of 0-1 linear constraints. We provide a first-of-kind cross-community evaluation of a selection of recently-proposed approaches applicable to MO-PBO, including first implementations of native MO-PBO algorithms we provide as well as approaches based on integer linear programming techniques and a translation-based approach to MO-MaxSAT, providing insights into the current state-of-the-art approaches to MO-PBO. In terms of algorithmic advances, we engineer MO-PBO solvers by harnessing recent advances in decision procedures for pseudo-Boolean constraints in order to lift multi-objective approaches recently developed for multi-objective optimization under propositional constraints (i.e., MO-MaxSAT) to the realm of MO-PBO. Extending on recent work on certified MO-MaxSAT solving, we also realize certified multi-objective pseudo-Boolean optimization by implementing proof logging for both our native MO-PBO approach and the translation-based MO-MaxSAT approach.

Keywords: Multi-objective optimization · pseudo-Boolean optimization · empirical evaluation · certified optimization

1 Introduction

The declarative approach—from mixed integer linear programming (ILP) [12] to finite-domain constraint optimization [64] and Boolean satisfiability (SAT) [8] based approaches including maximum satisfiability (MaxSAT) [3] together with its extensions to optimization modulo theories [62], and pseudo-Boolean optimization [66]—is key to efficiently solving various NP-hard combinatorial optimization problems arising from real-world settings. Various different algorithmic

* Work financially supported by Academy of Finland under grants 356046 and 362987. The authors thank the Finnish Computing Competence Infrastructure (FCCI) for computational and data storage resources.

techniques have been developed for the various declarative paradigms, ranging from linear programming based branch-and-cut algorithms standardly employed in ILP [12] to unsatisfiability-based search via iterative use of decision procedures in logic-based formalisms such as MaxSAT and its extensions [3]. This richness can be considered a virtue, with each paradigm offering its distinct features in terms of the constraint language and algorithmic approaches.

From MaxSAT to ILP, a majority of work on practical algorithms and their implementations towards developing increasingly effective declarative approaches to combinatorial optimization has focused on single-objective optimization problems. As real-world settings often intrinsically involve multiple conflicting objectives [29], beyond a plethora of heuristic approaches to multi-objective optimization [14], there has recently been interest in extending the reach of declarative approaches to enable efficiently solving multi-objective combinatorial optimization problems in various declarative paradigms, including ILP [42,25,34], finite-domain constraint optimization [55,67,10], and MaxSAT [68,15,48]. Some approaches address more restrictive settings, being geared to either e.g. bi-objective problems [10,48] or leximax optimization [11], while others do not restrict the number of objectives and in particular enable computing representative Pareto-optimal solutions [68,15] for every non-dominated point in the solution space—a task which is arguable noticeably more challenging than finding a representative optimal solution in the single-objective case.

In this work, we focus on engineering and evaluating practical approaches to solving multi-objective optimization problems expressed as pseudo-Boolean (PB) constraints [66], i.e., linear inequalities with integer coefficients over binary variables. Also known as 0-1 or binary linear constraints, PB constraints constitute on one hand a central fragment of integer programming, and on the other hand a natural generalization of conjunctive normal form clausal propositional constraints employed in SAT and MaxSAT. What makes multi-objective pseudo-Boolean optimization (MO-PBO) particularly interesting from the algorithmic perspective is that, firstly, recent approaches to multi-objective integer-linear programming [42,25,34] are directly applicable, and, secondly—as we will detail in this paper—recent advances in MO-MaxSAT can be harnessed in the context of MO-PBO either (i) by lifting multi-objective MaxSAT solving approaches [68,15,48] to obtain their native MO-PBO counterparts by employing recent advances in PB decision procedures [33,21,22], or (ii) by translating MO-PBO instances to MO-MaxSAT and employing MO-MaxSAT solvers. Contrasting the multi-objective ILP approaches, both of these two approaches allow for realizing—to the best of our knowledge for the first time—certified MO-PBO solving, i.e., integrating proof logging capabilities to MO-PBO solvers for obtaining certificates as guarantees for outputting exactly the sought-after solutions with reasonable overhead. The certificates are achieved by harnessing recent progress in certified MO-MaxSAT [47] and MaxSAT solving [72,6,45,5] via the VeriPB proof format [38,9]. Furthermore, we perform a first-of-kind cross-community evaluation of a wide selection of recently-proposed approaches applicable to MO-PBO, including the native MO-PBO algorithms we propose, the translation-based approach to MO-MaxSAT employing various recently-proposed MO-MaxSAT solvers, and approaches developed for multi-objective ILP solving. The results show that—both ones based on logical reasoning and those relying on classical ILP techniques—the three different approaches each have their role in contributing to the state of the art in MO-PBO solving. We provide open-source implementations of the MO-MaxSAT-based and native MO-PBO approaches, constituting the first certifying solvers for Pareto optimization under pseudo-Boolean constraints. The implementation together with benchmarks used, and empirical data reported on in this paper are available at

https://bitbucket.org/coreo-group/multi-objective-pbo.

2 Multi-objective Pseudo-Boolean Optimization

A literal ℓ is a $\{0, 1\}$-valued Boolean variable x or its negation $\overline{x}$. A normalized pseudo-Boolean (PB) constraint is a 0-1 linear inequality of form $C = (S \geq b)$, where $S = \sum_{i=1}^{k} c_i \ell_i$ is called a PB expression, c_i are positive integers, and b a non-negative integer often called the bound of the constraint. A PB formula is a conjunction of PB constraints $F = (C_1 \wedge C_2 \wedge \cdots \wedge C_N)$, often represented as a set of constraints.

An assignment α assigns a value in $\{0, 1\}$ to each variable x. An assignment α is extended to literals via $\alpha(\overline{x}) = 1 - \alpha(x)$ and further to PB expressions, constraints, and formulas, respectively, via

$$\alpha(S) = \sum_{i=1}^{k} c_i \alpha(\ell_i), \alpha(C) = \begin{cases} 1 \text{ if } \alpha(S) \geq b \\ 0 \text{ otherwise} \end{cases}, \ \alpha(F) = \min\{\alpha(C) \mid C \in F\}.$$

An assignment α for which $\alpha(F) = 1$ *satisfies* F, in which case α is a *solution* to F. When convenient, we view an assignment α as the set of literals α assigns to 1.

We write $r \Rightarrow (\sum_{i=1}^{k} c_i \ell_i \geq b)$ for the reified constraint $b\overline{r} + \sum_{i=1}^{k} c_i \ell_i \geq b$ expressing that the variable r implies the constraint $\sum_{i=1}^{k} c_i \ell_i \geq b$, and respectively $r \Leftarrow (\sum_{i=1}^{k} c_i \ell_i \geq b)$ for $Mr + \sum_{i=1}^{k} c_i \overline{\ell_i} \geq M$ where $M = \sum_{i=1}^{k} c_i - b + 1$ expressing that the constraint implies r.

A multi-objective pseudo-Boolean optimization (MO-PBO) instance consists of a formula F, and a tuple of p linear objective functions $(O_1, \ldots, O_p)$, represented as PB expressions. For two solutions α and β to F, α *dominates* β (in terms of Pareto optimality [29]) if $\alpha(O_i) \leq \beta(O_i)$ for all $i = 1, \ldots, p$, and $\alpha(O_i) < \beta(O_i)$ for some i. A solution α is *Pareto-optimal* if no other solution to F dominates α. We consider the MO-PBO task of finding the *non-dominated set*

$$\{(\alpha(O_1), \ldots, \alpha(O_p)) \mid \alpha \text{ is Pareto-optimal}\},$$

i.e., the objective values of all Pareto-optimal solutions, and typically one solution corresponding to each element in the non-dominated set. Note that this task is slightly different from finding all Pareto-optimal solutions, as there might be multiple solutions corresponding to the same element in the non-dominated set.

3 Relating MO-PBO to MO-MaxSAT and MO-ILP

Closely related to MO-PBO are multi-objective maximum satisfiability (MO-MaxSAT) and multi-objective integer linear programming (MO-ILP). In contrast to MO-PBO, in MO-MaxSAT all constraints are clauses, i.e., at-least-one constraints that have coefficients $c_i = 1$ and bound $b = 1$. MO-ILP differs from MO-PBO in that variables can take any integer values, with constraints taking the form $\sum_{i=1}^{k} c_i x_i \geq b$ with $c_i, b \in \mathbb{Z}$.

3.1 MO-MaxSAT

Employing an MO-MaxSAT algorithm to solve MO-PBO instances requires first encoding the PB constraints as clauses [28,49]. The encoded clausal instance can then be solved with an MO-MaxSAT solver to find the non-dominated set. In terms of practical approaches to MO-MaxSAT solving, Soh et al. [53,68] proposed an MO-MaxSAT algorithm based on enumeration of so-called P-minimal models. More recently, Cortes et al. [15] proposed a lower-bounding algorithm that uses P-minimal as a subroutine, Guerreiro et al. [41] extended on previous work employing minimal correction set enumeration to find Pareto-optimal solutions [71], and Jabs et al. [48] proposed BiOptSat, a MaxSAT-based algorithm applicable to bi-objective problems. In our work, we focus on the P-minimal and BiOptSat algorithm and discuss them in more detail in Section 4.

3.2 MO-ILP

As (MO-)PBO is a special case of (MO-)ILP, MO-ILP algorithms can be directly applied to solve MO-PBO instances. Various algorithms for MO-ILP have been recently proposed [42,70,18,69,36,63,25,35,34,4]. The main approaches to MO-ILP can be categorized into two classes: (i) branch-and-bound (B&B) search algorithms [52] and (ii) algorithms that solve a sequence of single-objective scalarizations of the MO problem [18].

Multi-objective B&B algorithms extend upper and lower bounds, used in single-objective B&B for pruning of search nodes, to so-called *upper and lower bound sets* [30]. Furthermore, the branching rules are extended with *objective / Pareto branching* [70], where the search space is split in objective space rather than variable space. Forget et al. [35] present a B&B framework for MO-ILP, which was later on extended with objective branching for any number of objectives [34].

Another approach to MO-ILP solving is based on single-objective scalarizations of the multi-objective problem [18,25]. Typically, optimal solutions to the single-objective scalarizations will be guaranteed to be Pareto-optimal with respect to the original problem instance. After finding one such optimal solution,

Input: $F, (O_1, \ldots, O_p)$
1 $F_{\mathrm{W}} \leftarrow F$ `sat`, $\alpha \leftarrow$ `Oracle`(F_W)
2 **while** `sat` **do**
3   **while** `sat` **do**
4         $F_{\mathrm{W}} \leftarrow F_{\mathrm{W}} \wedge \bigvee_{i=1}^{p} O_i < \alpha(O_i)$
5   `sat`, $\alpha \leftarrow$ `Oracle`$(F_W \wedge \bigwedge_{i=1}^{p} O_i \leq \alpha(O_i))$
6   **yield** α as Pareto-optimal
7   `sat`, $\alpha \leftarrow$ `Oracle`(F_W)

Algorithm 1: The P-minimal algorithm.

the algorithm will then split the search space into multiple regions, and solve another scalarization for each subregion. Dominguez-Rios et al. [25] improve on previous work in this line by proposing a new strategy of selecting which search region to explore next, and a new strategy of partitioning the search space when a solution is found.

4 Extending MO-MaxSAT Algorithms to PBO

As one of our contributions, we provide an open-source native MO-PBO level implementation of the state-of-the-art MO-MaxSAT algorithms P-minimal [53,68] and BiOptSat [48]. For this, we phrase the algorithms in terms of MO-PBO and explain how to adapt them to work on PB constraints natively by employing a decision procedure for PB constraints.

4.1 *P*-minimal

The P-minimal algorithm [53,68] for MO-MaxSAT can be viewed as multi-objective solution-improving search that iteratively queries a constraint oracle for a solution α to the constraints, and then restricts the search space with further constraints that exclude all solutions dominated by α. This continues until the non-dominated set is found.

Algorithm 1 details P-minimal. Here `Oracle`(F) denotes a query to a constraint oracle for a solution to the formula F. The query returns a Boolean `sat` indicating whether F has solutions, and one such solution α in the positive case. After initializing the working formula F_{W} to F and obtaining a solution α of F_{W}, the main loop of P-minimal (Lines 2–7 of Algorithm 1) iteratively adds a disjunction of constraints (in practice turned into a conjunction via reification, see details in next paragraph) that we call a Pareto dominance cut or PD cut for short, to block all solutions dominated by α to F_{W} on Line 4. Then it queries the oracle for a solution that dominates α on Line 5 by using temporary constraints that require the next solution to dominate α. When the oracle determines that there are no solutions that dominate α, α is guaranteed to be Pareto-optimal, and the search then continues by dropping the temporary constraints.

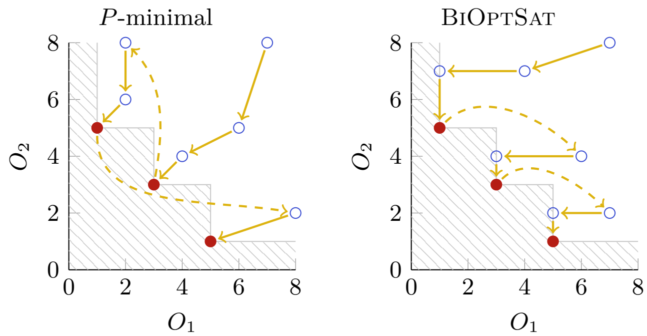

Fig. 1. Illustrations of the search trajectories of the P-minimal and BiOptSat algorithms.

Example 1. Let F and (O_1, O_2) constitute an MO-PBO instance with the non-dominated set $\{(1,5),(3,3),(5,1)\}$. Figure 1 illustrates this instance in objective space where the shaded area represents objective values for which no solutions exist. Assume that P-minimal is invoked on this instance and that on Line 1 we obtain α with $O_1(\alpha) = 7$ and $O_2(\alpha) = 8$. P-minimal now introduces a PD cut $(O_1 < 7 \vee O_2 < 8)$ and temporarily forces the next found solution to dominate α, by adding the constraints $O_1 \leq 7$ and $O_2 \leq 8$. By continuing the algorithm, we might find solutions with objective values $(6,5)$, $(4,4)$, and $(3,3)$, as illustrated in Figure 1 (left). After this, the oracle call on Line 5 returns false, and the temporary constraints are dropped. On Line 7 we might then find a solution with objective values $(2,8)$ (illustrated with a dashed arrow). A possible search trajectory discovering the entire non-dominated set before the algorithm terminates is illustrated on the left in Figure 1.

In the context of MO-MaxSAT, P-minimal is realized by instantiating `Oracle` as an incremental SAT-solver and encoding bound constraints $(O_i < \alpha(O_i))$ on the objectives as clauses. To lift P-minimal to MO-PBO, we use the pseudo-Boolean conflict-driven conflict learning [60] solver RoundingSat [33,21,22] as the oracle. The disjunctive constraint $\bigvee_{i=1}^{p} O_i < \alpha(O_i)$ is represented conjunctively as the reified constraints $r_i \Rightarrow (O_i < \alpha(O_i))$ and the clause $\sum_{i=1}^{p} r_i \geq 1$. The temporary constraints are realized via reified constraints and an incremental assumption interface [27,60] offered by RoundingSat out of the box. All in all, operating natively on the level of MO-PBO significantly simplifies the P-minimal algorithm compared to the case of MO-MaxSAT, as on the MO-PBO level there is no need to resort to (complex) clausal encodings of PB constraints.

4.2 BiOptSat

The BiOptSat framework [48] (see Algorithm 2) is specific to solving problems with two objectives, and works by enumerating the non-dominated points in increasing order for one objective and in decreasing order for the other. The

Input: $F, (O_1, O_2)$

1 $F_\mathrm{W} \leftarrow F$ `sat`, $\alpha \leftarrow$ `Oracle`(F_W)

2 **while** `sat` **do**

3 $\quad\alpha \leftarrow$ `Minimize`$(O_1 \textit{ s.t. } F_W)$

4 $\quad\alpha \leftarrow$ `Minimize`$(O_2 \textit{ s.t. } F_W \wedge O_1 \leq \alpha(O_1))$

5 $\quad$**yield** α as Pareto-optimal

6 $\quad F_\mathrm{W} \leftarrow F_\mathrm{W} \wedge O_2 < \alpha(O_2)$

7 $\quad$`sat`, $\alpha \leftarrow$ `Oracle`(F_W)

Algorithm 2: The BiOptSat framework.

approach first minimizes the first objective (Line 3), and then subsequently minimizes the second objective while fixing the value of the first (Line 4). Thereby, the obtained solution is guaranteed to be Pareto-optimal. The search then repeats after forcing the second objective to improve. This search strategy is also known as the lexicographic method [73,58].

Practical instantiations [48] of BiOptSat use solution-improving search for the minimization procedure on Line 4 but differ in how the minimization procedure over the first objective (Line 3) is instantiated. We focus on the variants based on Sat-Unsat (solution-improving search) and OLL [2,61,44] as the ones deemed most effective in previous work on MO-MaxSAT. In the Sat-Unsat variant, the oracle is queried for increasingly better solutions with respect to O_1 by temporarily adding the constraint $O_1 < O_1(\alpha)$ to the oracle, where α is the last-found solution. In contrast, in the OLL variant the oracle is queried for a solution setting all literals in O_1 to false. If such a solution does not exist, the oracle provides an explanation of the inconsistency, which is subsequently resolved by OLL and the objective reformulated. Both BiOptSat variants are implemented as described in previous work on solution-improving search and OLL for pseudo-Boolean optimization [23].

Example 2. Similarly to Example 1, we detail a possible search trajectory of BiOptSat in the Sat-Unsat variant on the right in Figure 1. In minimizing O_1 BiOptSat Sat-Unsat starts by obtaining a solution with objective values $(7, 8)$. Next, the oracle is queried again with the additional constraint $O_1 < 7$ added. This process might lead to first finding a solution at $(4, 7)$ and then one at $(1, 7)$, after which the oracle call with the additional constraint $O_1 < 1$ returns false. BiOptSat now continues to minimizing O_2 with the same solution-improving search procedure, while forcing the objective value of O_1 to remain minimal, but adding the constraint $O_1 \leq 1$. Once O_2 cannot be further minimized at $(1, 5)$, the solution is returned as Pareto-optimal, the constraints on O_1 are dropped, and the constraint $O_2 < 5$ is added to the oracle. The loop on Line 2 then starts over, and the entire instance is solved as illustrated in the right-hand side of Figure 1.

5 Certifying Pareto Optimality

Our second contribution concerns certified multi-objective pseudo-Boolean optimization. Proof logging—i.e., writing a machine-checkable proof of the reasoning steps performed by a solver, and afterwards verifying the reasoning steps made by the solver to obtain a guarantee on the correctness of the result produced—has become a readily-available feature in state-of-the-art SAT solvers [39,16,17] and has more recently been extended to single-objective MaxSAT [72,6,45,5] and single-objective PBO solvers [65]. The line of work on proof logging single-objective optimization is mainly based on the pseudo-Boolean VeriPB proof format [38,9]. In addition to a single objective, VeriPB supports certifying that the computed solutions are minimal with respect to a user-specified preorder over solutions. While originally proposed for proof logging symmetry and dominance breaking [9], the preorder in VeriPB has more recently been shown to generalize an objective in the sense of allowing VeriPB to be used to certify the correctness of a discovered non-dominated set in the multi-objective setting, without extending the proof system itself [47].

In this section, we detail how the same proof logging approach proposed for MO-MaxSAT [47] can be used for proof logging MO-PBO solvers, either based on a translation to the MaxSAT paradigm or by natively operating on PB constraints. For the sake of brevity, we will not explain the VeriPB proof system in detail, we refer the interested reader instead to previous work on VeriPB [38,9], especially in the multi-objective setting [47]. While early work on verifying results from integer programming solvers exists [13,43], we are not aware of any work on verifying MO-ILP results.

5.1 Proof Logging for MO-PBO via MO-MaxSAT

Since MO-PBO can be encoded as MO-MaxSAT and solved by existing MO-MaxSAT solvers, a natural approach toward proof-logging MO-PBO is to make use of the existing approaches to creating VeriPB proofs for various MO-MaxSAT algorithms, including P-minimal and BiOptSat [47]. However, this requires certifying the translation from an MO-PBO instance $(F, (O_1, \ldots, O_p))$ to a (clausal) MO-MaxSAT instance $(\texttt{As-Clauses}(F), (O_1, \ldots, O_p))$. Intuitively, the translation should guarantee that (i) any solution of F can be extended into a solution of $\texttt{As-Clauses}(F)$, and that (ii) any solution of $\texttt{As-Clauses}(F)$ is a solution of F. In practice, the certification of the translation is integrated into the proof produced by the certifying MO-MaxSAT solver, treating the translation of the PB constraints, in the same way as any other reasoning step of the solver. Together with certificates for the clausal reasoning steps of the MO-MaxSAT solvers described in [47], we end up with a single VeriPB proof that guarantees that at least one solution for each non-dominated point of the original MO-PBO instance has been found.

As a choice for encoding PB constraints to clauses for translating MO-PBO instances to MO-MaxSAT we employ the generalized totalizer encoding [49] as an often-employed encoding in MaxSAT literature. A generalized totalizer encoding can be viewed as a binary tree that has the literals in the PB constraint and their coefficients as leaves. The root of the tree and all internal nodes correspond to a set of auxiliary variables that—informally speaking—count the sum of the coefficients of the literals assigned to 1 at the leafs of the subtree rooted at that node. The PB constraint is enforced by a unit clause over the auxiliary variables at the root node that corresponds to its bound.

In order to certify the derivation of the clauses, we use a procedure originally described for totalizers [72] and later extended to the generalized totalizer encoding [47]. This procedure introduces the semantic definitions of the auxiliary variables at each node as auxiliary PB constraints in the proof and derives the clauses in the encoding from these definitions. However, the semantics in the proof are stricter than the produced encoding. Concretely, in the proof the output variables have semantics encoding *equality* to the PB expression exceeding a certain value, while the clauses in the generalized totalizer encoding only encode an *implication*. In order to avoid issues where a solution found in the MO-MaxSAT solver does not satisfy all constraints in the proof due to these differing semantics, we remove all auxiliary constraints from the VeriPB proof with the help of the derived deletion rule [38,9] once the clausal encoding for a PB constraint is generated.

5.2 Native Proof Logging for MO-PBO

The native MO-PBO approaches (recall Section 4) can also be extended to generate machine-checkable certificates. To do so, we use the VeriPB setup proposed in [47], encoding Pareto dominance as a preorder in the proof. In VeriPB format this preorder for objectives $(O_1, \ldots, O_p)$ is expressed as p constraints $O_i{\restriction_\alpha} \leq O_i{\restriction_\beta}$, where $O_i{\restriction_\alpha}$ is O_i under the assignment α. The formula formed by these constraints is true iff α dominates β or is equal to it. Since VeriPB verifies for redundant constraint that is added to the proof that a *witness* exists which is at least as good with respect to the preorder than any solutions the redundant constraint excludes, by loading the Pareto preorder as defined above, VeriPB ensures that Pareto-optimal solutions can only be excluded from the proof via the explicit solution-exclusion rule [47, Theorem 1].

Certifying a PD cut in the proof after finding the solution α is then done in the same way as in the MO-MaxSAT setting: First, a constraint that excludes all solutions that are dominated by α or have equal objective values (except for α itself) is added to the proof. This constraint is redundant, which can be justified using α as the witness, i.e., for any solution to the instance that does not satisfy this new constraint, α constitutes a solution that is at least as good with respect to Pareto optimality. Next, α is excluded from consideration with the help of the solution logging rule. Lastly, by combining the constraints from the previous steps, the PD cut itself can be derived.

Example 3. Recall the instance in Example 1. Assume that (i) $O_1 = 2x_2 + x_3 + 2x_4 + 3x_5$ and $O_2 = 3x_1 + x_2 + 2x_3 + x_4$, (ii) F includes no other variables, and (iii) P-minimal has found the Pareto-optimal solution $\alpha = \{x_1, \overline{x_2}, x_3, \overline{x_4}, \overline{x_5}\}$ for

Table 1. Example proof for certifying a PD cut.

ID	Pseudo-Boolean Constraint	Comment
	Input constraints and potential previous proof steps	
[a]	$8w_1 + 2\overline{x_2} + \overline{x_3} + 2\overline{x_4} + 3\overline{x_5} \geq 8$	witness: $\{w_1\}$
[b]	$\overline{w_1} + 2x_2 + x_3 + 2x_4 + 3x_5 \geq 1$	witness: $\{\overline{w_1}\}$
[c]	$3w_2 + 3\overline{x_1} + \overline{x_2} + 2\overline{x_3} + \overline{x_4} \geq 3$	witness: $\{w_2\}$
[d]	$5\overline{w_2} + 3x_1 + x_2 + 2x_3 + x_4 \geq 5$	witness: $\{\overline{w_2}\}$
[e]	$5\overline{w_1} + 5\overline{w_2} + x_1 + \overline{x_2} + x_3 + \overline{x_4} + \overline{x_5} \geq 5$	witness: $\alpha \cup \{w_1, w_2\}$
[f]	$\overline{x_1} + x_2 + \overline{x_3} + x_4 + x_5 \geq 1$	Log the solution α
[g]	$\overline{w_1} + \overline{w_2} \geq 1$	PD cut

which $(\alpha(O_1), \alpha(O_2)) = (1, 5)$. Table 1 shows example proof steps required for certifying a PD cut based on α. Steps [a], [b], [c], and [d] introduce new auxiliary variables w_1 and w_2 defined by $w_1 \Leftrightarrow O_1 \geq 1$ and $w_2 \Leftrightarrow O_2 \geq 5$ justified in the proof by witnesses that assign the (otherwise unconstrained) w_i variables the right way. With these definitions, the constraint introduced in step [e] is satisfied only by solutions that are not dominated by or equal to α, and α itself. Lastly, the solution α is ruled out with the solution logging rule (in step [f]) and the PD cut (expressed with the w variables) is derived in step [g] and justified by cutting planes reasoning as $([e] + [f])/5$.

By employing the described strategy for certifying PD cuts and a PB oracle which supports proof logging in VeriPB syntax (in our case RoundingSat), we implement proof logging for the algorithms described in Section 4. For proof logging P-minimal, only the PD cuts added to the oracle on Line 4 of Algorithm 1 need to be certified. For BiOptSat (Algorithm 2), the minimization on Line 3 is extended to derive the lower bound constraint $O_1 \geq \alpha(O_1)$ in the proof. When using solution-improving search, this constraint is derived by the oracle during the last unsatisfiable query, while for OLL, the proof logging procedure described in [6], adapted for the PB setting, is used. By combining the lower bound constraint $O_1 \geq \alpha(O_1)$ with a PD cut derived from α after Line 4, the constraint that is added to the working formula on Line 6 can be derived in the proof.

6 Empirical Evaluation

We turn to presenting results of a cross-community evaluation of the MaxSAT-translation based approach, native PB-based PBO approach and ILP-based approaches to MO-PBO. The experiments reportted on were run on 2.50-GHz Intel Xeon Gold 6248 machines with 381-GB RAM in RHEL under a per-instance 1-hour time and 32-GB memory limit.

6.1 Solvers

We implemented the PB-based MO-PBO algorithms proposed in Section 4 (P-minimal, and BiOptSat in the Sat-Unsat and OLL variants) in C++, using the PB solver RoundingSat [32,33,21,22] as the decision oracle. We also implemented proof logging for these algorithms, making use of VeriPB proof logging offered by the RoundingSat oracle, and extended the MO-MaxSAT solver Scuttle [46] by implementing certificates for the PBO-to-MaxSAT translation (recall Section 5), thereby obtaining certificates for the translation-based approach when conjoined with the MO-MaxSAT certificates implemented in Scuttle using its implementations for the P-minimal and the bi-objective BiOptSat algorithms.

As an additional recent MO-MaxSAT approach for the evaluation, we consider the core-guided approach from [15], which we refer to as lower-bounding (LB). In terms of MO-ILP, we consider the branch-and-bound approach from [34] (employing CPLEX v12.10 with non-trivial challenges in updating to a newer version) and the MultiObjectiveAlgorithms.jl (v1.3.5) [26] implementation of the scalarization-based algorithm [25], employing CPLEX (v20.1) as the ILP solver. Due to intrinsic ILP-specific numerical issues observed in preliminary experiments for the scalarization-based algorithm, we set the following parameters in CPLEX: absolute gap tolerance 10^{-6}, relative gap tolerance 0.0, and integrality tolerance 0.0. It should be noted, however, that even with these tuned parameters, we observed 2 instances where the scalarization-based ILP algorithm reports one more non-dominated point than the PBO and MaxSAT-based implementations of P-minimal, which are certified and thereby certifiably correct, underlining the need of proof logging to ensure correctness.

In the runtime comparison, we run all solvers without proof logging, and separately evaluate the proof logging overhead of the certified MO-MaxSAT and MO-PBO implementations.

6.2 Benchmarks and Setup

In our empirical evaluation we use multi-objective PBO benchmarks from seven problem domains: 365 bi-objective instances of learning interpretable decision rules (LIDR) [56] obtained from [48]; 388 bi-objective flying tourist problem (FTP) [59] instance obtained from [15] (after filtering out two empty instance files); 160 knapsack (KS) instances with 3–5 objectives and 100 assignment problem instances with 3 objectives, from [51,50] encoded as MO-PBO; and 35 bi-objective uncapacitated facility location problem (UFLP) instances from [37]. For further domains, we applied reverse-engineering described in [48] to the single-objective instances in the benchmark set of the Pseudo-Boolean Competition (2005–2024) [57,65], splitting multi-level objective combinations into individual objectives, keeping only benchmarks where reverse-engineering was succesful on all instances in the benchmark domain. With this process, we obtained 100 bi-objective haplotype inference [40] and 1513 development assurance level (DAL) [7,19] instances. The DAL instances were filtered based on the

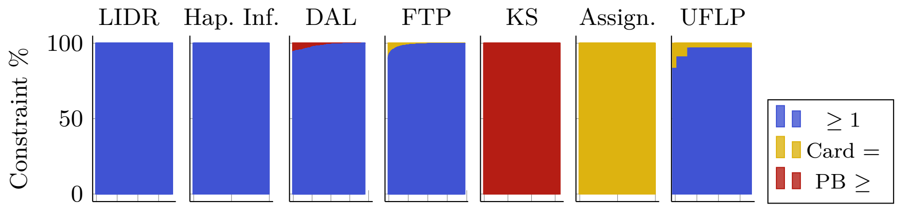

Fig. 2. Distribution of constraint types in the benchmark families.

LION9 challenge documentation [20] to remove duplicates differing only in the order of the objectives and ones turned from maximization to minimization by multiplying the objectives by -1, resulting in 378 distinct DAL instances.

Figure 2 shows the constraint type distribution of each benchmark domain, showing how the types of constraints vary significantly depending on the domain (from clausal at-least-ones through cardinality constraints to more generic PB constraints).

6.3 Results

Table 2 shows the number of solved instances for each solver per benchmark family, separated into multi-objective algorithms for arbitrary number of objectives and algorithm specific for bi-objective problems. All MaxSAT-based approaches (P-minimal, lower-bounding, and BiOptSat) perform well on the LIDR and haplotype inference benchmark instances which are based on fully clausal encodings. The ILP-based approaches perform well on the representatives of classical ILP problems: knapsack, assignment, and uncapacitated facility location. On the real-world MO-PBO instance families DAL and FTP, performance between the three constraint paradigms is not as clear-cut. The performance of the BiOptSat implementations compared to their repective P-minimal counterparts is relatively similar, showing the same trends with respect to the constraint

Table 2. Number of instances solved per benchmark family and solver.

		Multi-objective					Bi-objective		
		MaxSAT		PB	ILP		MaxSAT	PB	
Family	Total	P-min	LB	P-min	B&B	Scalar	BOS-SU	BOS-SU	BOS-OLL
LIDR	365	216	202	162	148	199	**221**	169	179
Hap. Inf.	100	**20**	18	18	0	6	19	17	18
DAL	378	216	197	**251**	154	159		—	
FTP	388	112	122	146	123	**257**	113	145	117
KS	160	62	58	85	114	**151**		—	
Assign.	100	19	20	18	51	**55**		—	
UFLP	35	1	3	12	**30**	29	2	12	2

Table 3. Number of uniquely solved instances and virtual best solver contribution per benchmark family for SCUTTLE P-minimal (MS), native PBO P-minimal (PBO), and the scalarization-based approach (ILP).

			Uniquely solved			VBS contribution		
Family	Total	VBS solved	MaxSAT	PBO	ILP	MaxSAT	PBO	ILP
LIDR	365	222	**21**	0	6	**200**	10	12
Hap. Inf.	100	20	**2**	0	0	**20**	0	0
DAL	378	260	9	**43**	0	85	**169**	6
FTP	388	258	0	0	**108**	1	115	**141**
KS	160	158	4	1	**64**	7	65	**84**
Assign.	100	59	0	0	**35**	0	10	**45**
UFLP	35	29	0	0	**17**	0	3	**26**

paradigms: with the introduction of non-clausal constraints the PB-based implementations outperform the MaxSAT-based implementations, but fail to fully match the performance of the ILP-based algorithms.

Towards a more fine-grained performance analysis, we select one approach as representative from each paradigm: for MO-MaxSAT SCUTTLE P-minimal, for MO-PBO, the native P-minimal implementation, and for MO-ILP the scalarization-based approach [25]. For the three selected representatives Table 3 shows the number of uniquely solved instances, the number of contributions to the virtual best solver (VBS)—i.e., the number of instances that a given solver solved the fastest out of the selected three, as well as the number of solved instances solved by the VBS. Additionally, Figure 3 shows a per-instance runtime comparison of the approaches from the three paradigms. The trends visible in Table 2 can be seen again: the MO-MaxSAT-based approach performs best on clausal instances, while the ILP-based approach performs best on classical ILP problems. We observe that the native PBO approach significantly outperforms the others on the DAL domain, contributing to the virtual best solver more than twice as much as MO-MaxSAT approach, whereas the contribution of ILP on the VBS on the DAL domain is very small without any uniquely solved instances. On the FTP domain, even though the ILP approach performs the best, we observe an almost equal VBS contribution from the native PBO approach. The same holds for the knapsack domain. The pairwise runtime comparison in Figure 3 further corroborates the complementary nature of the three approaches.

Complementary to the runtime performance, the ILP-based approach may suffer from numerical issues as also observed in our experiments. Our first-of-kind certified MO-PBO solvers based on translation to MO-MaxSAT and on the other hand liftings of recent MO-MaxSAT algorithms to natively work on MO-PBO offer guaranteed correctness. We also evaluated the cost of obtaining these certificates: with proof logging enabled, we observed relatively modest average overheads of 25% (for native MO-PBO P-minimal) and 48% (for SCUTTLE P-minimal) compared to running the solvers with proof logging disabled.

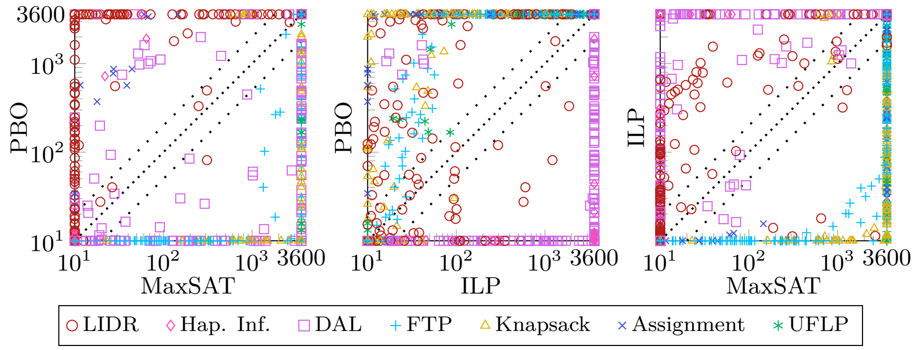

Fig. 3. Per-instance solving time comparisons representative solvers for each paradigm.

7 Conclusions

We engineered first-of-kind certifying algorithms and evaluated a range of approaches for pseudo-Boolean optimization. The results of the cross-community evaluation of the approaches show that MO-ILP, native MO-PBO, and translation-based MO-MaxSAT approaches offer complementary performance. On problems yielding mostly at-most-one constraints translating to MO-MaxSAT is competitive. For instances (with even relatively small number of) more generic PB constraints, it appears beneficial often to employ techniques which natively search on such constraints. For problem settings for which ILP solvers have been classically employed, MO-ILP appears also a good choice. Complementary to the runtime performance, our implementations of MO-PBO solvers based on translation to MO-MaxSAT and on liftings of recent MO-MaxSAT algorithms to natively work on MO-PBO offer guaranteed correctness via proof logging. Empirical runtime overhead from proof logging for these approaches is relatively minor, especially when compared to significant overheads reported for numerically exact ILP solvers in the single objective setting [31].

References

1. Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2–9, 2021. AAAI Press (2021), https://www.aaai.org/Library/AAAI/aaai21contents.php
2. Andres, B., Kaufmann, B., Matheis, O., Schaub, T.: Unsatisfiability-based optimization in clasp. In: Dovier, A., Costa, V.S. (eds.) Technical Communications of the 28th International Conference on Logic Programming, ICLP 2012, September 4–8, 2012, Budapest, Hungary. LIPIcs, vol. 17, pp. 211–221. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2012). https://doi.org/10.4230/LIPIcs.ICLP.2012.211, http://drops.dagstuhl.de/opus/portals/extern/index.php?semnr=12008

3. Bacchus, F., Järvisalo, M., Martins, R.: Maximum satisfiability. In: Biere et al. [8], pp. 929–991. https://doi.org/10.3233/FAIA201008
4. Bauß, J., Parragh, S.N., Stiglmayr, M.: On improvements of multi-objective branch and bound. EURO Journal on Computational Optimization **12**, 100099 (2024). https://doi.org/10.1016/j.ejco.2024.100099
5. Berg, J., Bogaerts, B., Nordström, J., Oertel, A., Paxian, T., Vandesande, D.: Certifying without loss of generality reasoning in solution-improving maximum satisfiability. In: Shaw, P. (ed.) 30th International Conference on Principles and Practice of Constraint Programming, CP 2024, September 2–6, 2024, Girona, Spain. LIPIcs, vol. 307, pp. 4:1–4:28. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2024). https://doi.org/10.4230/LIPIcs.CP.2024.4, https://www.dagstuhl.de/dagpub/978-3-95977-336-2
6. Berg, J., Bogaerts, B., Nordström, J., Oertel, A., Vandesande, D.: Certified core-guided MaxSAT solving. In: Pientka, B., Tinelli, C. (eds.) Automated Deduction—CADE 29—29th International Conference on Automated Deduction, Rome, Italy, July 1–4, 2023, Proceedings. Lecture Notes in Computer Science, vol. 14132, pp. 1–22. Springer (2023). https://doi.org/10.1007/978-3-031-38499-8_1
7. Bieber, P., Delmas, R., Seguin, C.: DALculus—theory and tool for development assurance level allocation. In: Flammini, F., Bologna, S., Vittorini, V. (eds.) Computer Safety, Reliability, and Security—30th International Conference, SAFECOMP 2011, Naples, Italy, September 19–22, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6894, pp. 43–56. Springer (2011). https://doi.org/10.1007/978-3-642-24270-0_4
8. Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.): Handbook of Satisfiability—Second Edition, Frontiers in Artificial Intelligence and Applications, vol. 336. IOS Press (2021). https://doi.org/10.3233/FAIA336
9. Bogaerts, B., Gocht, S., McCreesh, C., Nordström, J.: Certified dominance and symmetry breaking for combinatorial optimisation. Journal of Artificial Intelligence Research **77**, 1539–1589 (2023). https://doi.org/10.1613/jair.1.14296
10. Buchet, S., Allouche, D., de Givry, S., Schiex, T.: Bi-objective discrete graphical model optimization. In: Dilkina [24], pp. 136–152. https://doi.org/10.1007/978-3-031-60597-0_10
11. Cabral, M., Janota, M., Manquinho, V.: SAT-based leximax optimisation algorithms. In: Meel, K.S., Strichman, O. (eds.) 25th International Conference on Theory and Applications of Satisfiability Testing, SAT 2022, August 2–5, 2022, Haifa, Israel. LIPIcs, vol. 236, pp. 29:1–29:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2022). https://doi.org/10.4230/LIPIcs.SAT.2022.29, https://www.dagstuhl.de/dagpub/978-3-95977-242-6
12. Chen, D., Batson, R.G., Dang, Y.: Applied Integer Programming: Modeling and Solution. Wiley (Dec 2009). https://doi.org/10.1002/9781118166000, http://dx.doi.org/10.1002/9781118166000
13. Cheung, K.K.H., Gleixner, A.M., Steffy, D.E.: Verifying integer programming results. In: Eisenbrand, F., Könemann, J. (eds.) Integer Programming and Combinatorial Optimization—19th International Conference, IPCO 2017, Waterloo, ON, Canada, June 26–28, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10328, pp. 148–160. Springer (2017). https://doi.org/10.1007/978-3-319-59250-3_13
14. Coello, C.A.C.: Multi-objective optimization. In: Martí, R., Pardalos, P.M., Resende, M.G.C. (eds.) Handbook of Heuristics., pp. 177–204. Springer (2018). https://doi.org/10.1007/978-3-319-07124-4_17

15. Cortes, J., Lynce, I., Manquinho, V.: New core-guided and hitting set algorithms for multi-objective combinatorial optimization. In: Sankaranarayanan, S., Sharygina, N. (eds.) Tools and Algorithms for the Construction and Analysis of Systems—29th International Conference, TACAS 2023, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Paris, France, April 22–27, 2023, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13994, pp. 55–73. Springer (2023). https://doi.org/10.1007/978-3-031-30820-8_7
16. Cruz-Filipe, L., Heule, M.J.H., Jr., W.A.H., Kaufmann, M., Schneider-Kamp, P.: Efficient certified RAT verification. In: de Moura, L. (ed.) Automated Deduction—CADE 26—26th International Conference on Automated Deduction, Gothenburg, Sweden, August 6–11, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10395, pp. 220–236. Springer (2017). https://doi.org/10.1007/978-3-319-63046-5_14
17. Cruz-Filipe, L., Marques-Silva, J., Schneider-Kamp, P.: Efficient certified resolution proof checking. In: Legay, A., Margaria, T. (eds.) Tools and Algorithms for the Construction and Analysis of Systems—23rd International Conference, TACAS 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22–29, 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10205, pp. 118–135 (2017). https://doi.org/10.1007/978-3-662-54577-5_7
18. Dächert, K., Klamroth, K.: A linear bound on the number of scalarizations needed to solve discrete tricriteria optimization problems. Journal of Global Optimization **61**, 643–676 (2015). https://doi.org/10.1007/s10898-014-0205-z
19. Delmas, K., Chambert, L., Frazza, C., Seguin, C.: Optimization of development assurance level allocation. In: 2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC). pp. 1–10 (2023). https://doi.org/10.1109/DASC58513.2023.10311260
20. Delmas, R., Seguin, C., Bieber, P.: DALculus optimization benchmarks. https://www.cril.univ-artois.fr/ChallengeLion9/Dalculus.pdf
21. Devriendt, J.: Watched propagation of 0-1 integer linear constraints. In: Simonis, H. (ed.) Principles and Practice of Constraint Programming—26th International Conference, CP 2020, Louvain-la-Neuve, Belgium, September 7–11, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12333, pp. 160–176. Springer (2020). https://doi.org/10.1007/978-3-030-58475-7_10
22. Devriendt, J., Gleixner, A.M., Nordström, J.: Learn to relax: Integrating 0-1 integer linear programming with pseudo-boolean conflict-driven search. Constraints—An International Journal **26**, 26–55 (2021). https://doi.org/10.1007/s10601-020-09318-x
23. Devriendt, J., Gocht, S., Demirovic, E., Nordström, J., Stuckey, P.J.: Cutting to the core of pseudo-boolean optimization: Combining core-guided search with cutting planes reasoning. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2–9, 2021 [1], pp. 3750–3758. https://doi.org/10.1609/aaai.v35i5.16492, https://www.aaai.org/Library/AAAI/aaai21contents.php
24. Dilkina, B. (ed.): Integration of Constraint Programming, Artificial Intelligence, and Operations Research—21st International Conference, CPAIOR 2024, Uppsala, Sweden, May 28–31, 2024, Proceedings, Part I, Lecture Notes in Computer Science, vol. 14742. Springer (2024). https://doi.org/10.1007/978-3-031-60597-0

25. Domínguez-Ríos, M.Á., Chicano, F., Alba, E.: Effective anytime algorithm for multiobjective combinatorial optimization problems. Information Sciences **565**, 210–228 (2021). https://doi.org/10.1016/j.ins.2021.02.074
26. Dowson, O.: MultiObjectiveAlgorithms.jl. https://github.com/jump-dev/MultiObjectiveAlgorithms.jl
27. Eén, N., Sörensson, N.: Temporal induction by incremental SAT solving. In: Strichman, O., Biere, A. (eds.) First International Workshop on Bounded Model Checking, BMC@CAV 2003, Boulder, Colorado, USA, July 13, 2003. Electronic Notes in Theoretical Computer Science, vol. 89, pp. 543–560. Elsevier (2003). https://doi.org/10.1016/S1571-0661(05)82542-3, https://www.sciencedirect.com/journal/electronic-notes-in-theoretical-computer-science/vol/89/issue/4
28. Eén, N., Sörensson, N.: Translating pseudo-boolean constraints into SAT. Journal on Satisfiability, Boolean Modeling and Computation **2**, 1–26 (2006). https://doi.org/10.3233/sat190014
29. Ehrgott, M.: Multicriteria Optimization (2. ed.). Springer (2005). https://doi.org/10.1007/3-540-27659-9
30. Ehrgott, M., Gandibleux, X.: Bound sets for biobjective combinatorial optimization problems. Computers & Operations Research **34**, 2674–2694 (2007). https://doi.org/10.1016/j.cor.2005.10.003
31. Eifler, L., Gleixner, A.M.: A computational status update for exact rational mixed integer programming. Mathematical Programming **197**, 793–812 (2023). https://doi.org/10.1007/s10107-021-01749-5
32. Elffers, J., Devriendt, J., Gocht, S., Nordström, J.: RoundingSat: The pseudo-Boolean solver powered by proof complexity. https://gitlab.com/MIAOresearch/software/roundingsat
33. Elffers, J., Nordström, J.: Divide and conquer: Towards faster pseudo-boolean solving. In: Lang [54], pp. 1291–1299. https://doi.org/10.24963/ijcai.2018/180, http://www.ijcai.org/proceedings/2018/
34. Forget, N., Gadegaard, S.L., Klamroth, K., Nielsen, L.R., Przybylski, A.: Branch-and-bound and objective branching with three or more objectives. Computers & Operations Research **148**, 106012 (2022). https://doi.org/10.1016/j.cor.2022.106012
35. Forget, N., Gadegaard, S.L., Nielsen, L.R.: Warm-starting lower bound set computations for branch-and-bound algorithms for multi objective integer linear programs. European Journal of Operational Research **302**, 909–924 (2022). https://doi.org/10.1016/j.ejor.2022.01.047
36. Gadegaard, S.L., Nielsen, L.R., Ehrgott, M.: Bi-objective branch-and-cut algorithms based on LP relaxation and bound sets. INFORMS Journal on Computing **31**, 790–804 (2019). https://doi.org/10.1287/ijoc.2018.0846
37. Gandibleux, X., Contibutors: vOptLib: Library of numerical instances (MOMIP, MOLP, MOIP, MOCO). https://github.com/vOptSolver/vOptLib
38. Gocht, S., Nordström, J.: Certifying parity reasoning efficiently using pseudo-boolean proofs. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2–9, 2021 [1], pp. 3768–3777. https://doi.org/10.1609/aaai.v35i5.16494, https://www.aaai.org/Library/AAAI/aaai21contents.php
39. Goldberg, E.I., Novikov, Y.: Verification of proofs of unsatisfiability for CNF formulas. In: 2003 Design, Automation and Test in Europe Conference and Exposition

(DATE 2003), 3–7 March 2003, Munich, Germany. pp. 10886–10891. IEEE Computer Society (2003), https://doi.ieeecomputersociety.org/10.1109/DATE.2003.10008

40. Graça, A., Lynce, I., Marques-Silva, J., Oliveira, A.L.: Efficient and accurate haplotype inference by combining parsimony and pedigree information. In: Horimoto, K., Nakatsui, M., Popov, N. (eds.) Algebraic and Numeric Biology—4th International Conference, ANB 2010, Hagenberg, Austria, July 31- August 2, 2010, Revised Selected Papers. Lecture Notes in Computer Science, vol. 6479, pp. 38–56. Springer (2010). https://doi.org/10.1007/978-3-642-28067-2_3
41. Guerreiro, A.P., Cortes, J., Vanderpooten, D., Bazgan, C., Lynce, I., Manquinho, V., Figueira, J.R.: Exact and approximate determination of the pareto front using minimal correction subsets. Computers & Operations Research **153**, 106153 (2023). https://doi.org/10.1016/j.cor.2023.106153
42. Halffmann, P., Schäfer, L.E., Dächert, K., Klamroth, K., Ruzika, S.: Exact algorithms for multiobjective linear optimization problems with integer variables: A state of the art survey. Journal of Multi-Criteria Decision Analysis **29**(5-6), 341–363 (2022). https://doi.org/10.1002/mcda.1780, https://onlinelibrary.wiley.com/doi/abs/10.1002/mcda.1780
43. Hoen, A., Oertel, A., Gleixner, A.M., Nordström, J.: Certifying MIP-based presolve reductions for 0-1 integer linear programs. In: Dilkina [24], pp. 310–328. https://doi.org/10.1007/978-3-031-60597-0_20
44. Ignatiev, A., Morgado, A., Marques-Silva, J.: RC2: an efficient MaxSAT solver. Journal on Satisfiability, Boolean Modeling and Computation **11**, 53–64 (2019). https://doi.org/10.3233/SAT190116
45. Ihalainen, H., Oertel, A., Tan, Y.K., Berg, J., Järvisalo, M., Myreen, M.O., Nordström, J.: Certified MaxSAT preprocessing. In: Benzmüller, C., Heule, M.J.H., Schmidt, R.A. (eds.) Automated Reasoning—12th International Joint Conference, IJCAR 2024, Nancy, France, July 3–6, 2024, Proceedings, Part I. Lecture Notes in Computer Science, vol. 14739, pp. 396–418. Springer (2024). https://doi.org/10.1007/978-3-031-63498-7_24
46. Jabs, C.: Scuttle: A multi-objective MaxSAT solver. https://bitbucket.org/coreo-group/scuttle
47. Jabs, C., Berg, J., Bogaerts, B., Järvisalo, M.: Certifying pareto optimality in multi-objective maximum satisfiability. In: Gurfinkel, A., Heule, M. (eds.) Tools and Algorithms for the Construction and Analysis of Systems—31st International Conference, TACAS 2025, Held as Part of the International Joint Conferences on Theory and Practice of Software, ETAPS 2025, Hamilton, ON, Canada, May 3–8, 2025, Proceedings, Part II. Lecture Notes in Computer Science, vol. 15697, pp. 108–129. Springer (2025). https://doi.org/10.1007/978-3-031-90653-4_6
48. Jabs, C., Berg, J., Niskanen, A., Järvisalo, M.: From single-objective to bi-objective maximum satisfiability solving. Journal of Artificial Intelligence Research **80**, 1223–1269 (2024). https://doi.org/10.1613/jair.1.15333
49. Joshi, S., Martins, R., Manquinho, V.: Generalized totalizer encoding for pseudo-boolean constraints. In: Pesant, G. (ed.) Principles and Practice of Constraint Programming—21st International Conference, CP 2015, Cork, Ireland, August 31 – September 4, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9255, pp. 200–209. Springer (2015). https://doi.org/10.1007/978-3-319-23219-5_15
50. Kirlik, G.: https://web.archive.org/web/20240517210648/http://home.ku.edu.tr/~moolibrary/

51. Kirlik, G., Sayin, S.: A new algorithm for generating all nondominated solutions of multiobjective discrete optimization problems. European Journal of Operational Research **232**, 479–488 (2014). https://doi.org/10.1016/j.ejor.2013.08.001
52. Kiziltan, G., Yucaoğlu, E.: An algorithm for multiobjective zero-one linear programming. Management Science **29**(12), 1444–1453 (Dec 1983). https://doi.org/10.1287/mnsc.29.12.1444
53. Koshimura, M., Nabeshima, H., Fujita, H., Hasegawa, R.: Minimal model generation with respect to an atom set. In: Peltier, N., Sofronie-Stokkermans, V. (eds.) Proceedings of the 7th International Workshop on First-Order Theorem Proving, FTP 2009, Oslo, Norway, July 6–7, 2009. CEUR Workshop Proceedings, vol. 556. CEUR-WS.org (2009), https://ceur-ws.org/Vol-556/paper06.pdf
54. Lang, J. (ed.): Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13–19, 2018, Stockholm, Sweden. ijcai.org (2018), http://www.ijcai.org/proceedings/2018/
55. Malalel, S., Malapert, A., Pelleau, M., Régin, J.C.: MDD archive for boosting the pareto constraint. In: Yap [74], pp. 24:1–24:15. https://doi.org/10.4230/LIPIcs.CP.2023.24, https://www.dagstuhl.de/dagpub/978-3-95977-300-3
56. Malioutov, D., Meel, K.S.: MLIC: A MaxSAT-based framework for learning interpretable classification rules. In: Hooker, J.N. (ed.) Principles and Practice of Constraint Programming—24th International Conference, CP 2018, Lille, France, August 27–31, 2018, Proceedings. Lecture Notes in Computer Science, vol. 11008, pp. 312–327. Springer (2018). https://doi.org/10.1007/978-3-319-98334-9_21
57. Manquinho, V.M., Roussel, O.: The first evaluation of pseudo-boolean solvers (PB'05). Journal on Satisfiability, Boolean Modeling and Computation **2**, 103–143 (2006). https://doi.org/10.3233/sat190018
58. Marler, R., Arora, J.: Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization **26**, 369–395 (04 2004). https://doi.org/10.1007/s00158-003-0368-6
59. Marques, R., Russo, L.M.S., Roma, N.: Flying tourist problem: Flight time and cost minimization in complex routes. Expert Systems with Applications **130**, 172–187 (2019). https://doi.org/10.1016/j.eswa.2019.04.024
60. Marques-Silva, J., Lynce, I., Malik, S.: Conflict-driven clause learning SAT solvers. In: Biere et al. [8], pp. 133–182. https://doi.org/10.3233/FAIA200987
61. Morgado, A., Dodaro, C., Marques-Silva, J.: Core-guided MaxSAT with soft cardinality constraints. In: O'Sullivan, B. (ed.) Principles and Practice of Constraint Programming—20th International Conference, CP 2014, Lyon, France, September 8–12, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8656, pp. 564–573. Springer (2014). https://doi.org/10.1007/978-3-319-10428-7_41
62. Nieuwenhuis, R., Oliveras, A.: On SAT modulo theories and optimization problems. In: Biere, A., Gomes, C.P. (eds.) Theory and Applications of Satisfiability Testing—SAT 2006, 9th International Conference, Seattle, WA, USA, August 12–15, 2006, Proceedings. Lecture Notes in Computer Science, vol. 4121, pp. 156–169. Springer (2006). https://doi.org/10.1007/11814948_18
63. Parragh, S.N., Tricoire, F.: Branch-and-bound for bi-objective integer programming. INFORMS Journal on Computing **31**, 805–822 (2019). https://doi.org/10.1287/ijoc.2018.0856
64. Rossi, F., van Beek, P., Walsh, T. (eds.): Handbook of Constraint Programming, Foundations of Artificial Intelligence, vol. 2. Elsevier (2006), https://www.sciencedirect.com/science/bookseries/15746526/2
65. Roussel, O.: Pseudo-Boolean Competition 2024. https://www.cril.univ-artois.fr/PB24/

66. Roussel, O., Manquinho, V.: Pseudo-boolean and cardinality constraints. In: Biere et al. [8], pp. 1087–1129. https://doi.org/10.3233/FAIA201012
67. Roussel, S., Polacsek, T., Chan, A.: Assembly line preliminary design optimization for an aircraft. In: Yap [74], pp. 32:1–32:19. https://doi.org/10.4230/LIPIcs.CP.2023.32, https://www.dagstuhl.de/dagpub/978-3-95977-300-3
68. Soh, T., Banbara, M., Tamura, N., Le Berre, D.: Solving multiobjective discrete optimization problems with propositional minimal model generation. In: Beck, J.C. (ed.) Principles and Practice of Constraint Programming—23rd International Conference, CP 2017, Melbourne, VIC, Australia, August 28 – September 1, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10416, pp. 596–614. Springer (2017). https://doi.org/10.1007/978-3-319-66158-2_38
69. Stidsen, T.R., Andersen, K.A.: A hybrid approach for biobjective optimization. Discrete Optimization **28**, 89–114 (2018). https://doi.org/10.1016/j.disopt.2018.02.001
70. Stidsen, T.R., Andersen, K.A., Dammann, B.: A branch and bound algorithm for a class of biobjective mixed integer programs. Management Science **60**, 1009–1032 (2014). https://doi.org/10.1287/mnsc.2013.1802
71. Terra-Neves, M., Lynce, I., Manquinho, V.: Multi-objective optimization through pareto minimal correction subsets. In: Lang [54], pp. 5379–5383. https://doi.org/10.24963/ijcai.2018/757, http://www.ijcai.org/proceedings/2018/
72. Vandesande, D., Wulf, W.D., Bogaerts, B.: QMaxSATpb: A certified MaxSAT solver. In: Gottlob, G., Inclezan, D., Maratea, M. (eds.) Logic Programming and Nonmonotonic Reasoning—16th International Conference, LPNMR 2022, Genova, Italy, September 5–9, 2022, Proceedings. Lecture Notes in Computer Science, vol. 13416, pp. 429–442. Springer (2022). https://doi.org/10.1007/978-3-031-15707-3_33
73. Wassenhove, L.N.V., Gelders, L.F.: Solving a bicriterion scheduling problem. European Journal of Operational Research **4**(1), 42–48 (jan 1980). https://doi.org/10.1016/0377-2217(80)90038-7
74. Yap, R.H.C. (ed.): 29th International Conference on Principles and Practice of Constraint Programming, CP 2023, August 27–31, 2023, Toronto, Canada, LIPIcs, vol. 280. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2023), https://www.dagstuhl.de/dagpub/978-3-95977-300-3