An experimental evaluation of Max-SAT and PB solvers on over-subscription planning problems

Marco Maratea

DIST, University of Genova, Viale F. Causa 15, Genova, Italy.
marco@dist.unige.it

Abstract

We present an evaluation of Max-SAT and Pseudo-Boolean (PB) solvers on a novel and interesting application domain involving planning problems with preferences expressed on actions preconditions and/or goals. These are over-subscription planning problems, i.e., planning problems in which not all the goals can be satisfied, thus practically very important, where a cost is associated to the violation of goals and/or actions preconditions, which include all domains from the "SimplePreferences" track of the 5th International Planning Competition (IPC-5). Such benchmarks are reduced to Max-SAT and PB problems, which provide two very natural ways to express this situation. We run a wide experimental analysis involving all best performing Max-SAT and PB solvers, and all the domains from the "SimplePreferences" track of the IPC-5. Our analysis reveals what are the solvers that, at the moment, perform best on these benchmarks, and identifies, at the same time, challenging Max-SAT and PB benchmarks that we plan to submit to the next evaluations.

1 Introduction

Max-SAT and Pseudo-Boolean (PB) problems are two extensions of the well-known propositional satisfiability (SAT) problem. The goal of these two extensions is to deal with optimization problems where costs, or weights, are associated to the satisfaction/violation of clauses and/or variables of the problem. These formalisms allow an end user to naturally reason with integer numbers, which is one of the main limitation of SAT, instead of relying on (somewhat complicated and/or space consuming) encodings, e.g., [1, 2]. Given a formula φ (in CNF, i.e., a set of clauses, a clause being a set of literals), in the "classical" Max-SAT problem the goal is to find an assignment to the variables in φ that maximizes the number of satisfied clauses; in its partial variant there are both "hard" and "soft" clauses, and the goal is to find an assignment that satisfies all hard clauses and as many of the soft as possible. In a (linear) PB problem, an optimization function is defined over the variables of the problem, which consist of a set of PB constraints. A PB constraint is an extension of a SAT clause having integer coefficients and a bound on the "value" that the constraint can have, where variable's truth is interpreted as 0/1. In the linear case, PB problems correspond to 0-1 Integer Programming (IP) problems, i.e., IP problems where variables are indeed interpreted as 0/1. Thanks

Proceedings of the 17th International RCRA workshop (RCRA 2010):
Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion
Bologna, Italy, June 10–11, 2010

to a series of evaluations [3, 4][1], a number of benchmarks and efficient systems are now available.

In this paper we present an evaluation of Max-SAT and Pseudo-Boolean (PB) solvers on a novel and interesting application domain involving planning problems with preferences expressed on actions preconditions and/or goals. These are over-subscription planning problems [5, 6], i.e., planning problems in which not all the goals can be satisfied, thus practically very important, where a cost is associated to the violation of goals and/or actions preconditions, which include all domains from the "SimplePreferences" track of the 5th International Planning Competition (IPC-5)[2] [7]. Considering a fixed plan horizon (i.e., makespan), such benchmarks are reduced to Max-SAT and PB problems, which provide two very natural ways to express this situation. The reduction is done in two steps: (i) non-STRIPS problems of the IPC-5 are translated into STRIPS [8] problems, using state-of-the-art techniques [9, 10] and tools (ADL2STRIPS [11]); and (ii) a modified version of the famous SATPLAN planner [12, 13] is run, at fixed makespan, on the resulting STRIPS problems to generate Max-SAT and PB instances. Specifically, in the first case the approach generates a Weighted Partial Max-SAT problem, which is a further extension of the Max-SAT problem in which a positive integer weight is associated to each soft clause, and the goal is to satisfy all hard clauses and maximize the sum of weights associated to satisfied soft clauses; in the second, PB constraints correspond to SAT clauses. We have to note that a modeling of planning problems with preferences, expressed through the PDDL3 [14, 7] language, in 0-1 IP has been already presented in [15]; however, no implementation and experimental analysis have been provided.[3]

We run a wide experimental analysis involving all best performing Max-SAT and PB solvers, and all domains in the "SimplePreferences" track of the IPC-5, which include (quantitative) preferences expressed on goals and/or actions preconditions. Our analysis reveals what are the solvers that, at the moment, perform best on these benchmarks: results are often mixed considering different domains, but the best overall solver is MINISAT+ [16], a PB solver that relies on a compilation into a series of SAT problems, which is able to solve the highest number of benchmarks among the ones presented, usually in "short" time. At the same time, our analysis identifies challenging Max-SAT and PB benchmarks that we plan to submit to the next evaluations.

2 Modeling and Implementation

In the section we present in details how we have modeled the problem of interest, that we have highlighted in the previous section. We have evaluated the domains

[1] See `http://www.maxsat.udl.cat/09/` and `http://www.cril.univ-artois.fr/PB09/` for the last.

[2] `http://zeus.ing.unibs.it/ipc-5/`.

[3] This is confirmed by recent personal communications with Menkes van der Briel.

from the "SimplePreferences" track of the IPC-5 where plan metrics, in terms of quantitative preferences, are expressed on goals and/or on actions preconditions, i.e., the Pathways, Storage, Trucks, Openstacks and TPP domains. Given that SATPLAN can only handle STRIPS domains, while such domains are non-STRIPS, and some ADL [17] constructs are used, we have first adapted the non-STRIPS problems in the following way:

- the preferences[4] expressed on actions preconditions are treated as follows: each action containing such preference is expressed with two actions that do not contain preferences. For both actions, the related preference formula is treated as hard, further negated in the second. The second action also achieves a new dummy literal; and
- the goal preferences in the IPC-5 problems are translated into preconditions of dummy actions, which achieve new dummy literals defining the new problem goals.

The treatment of actions preferences is inspired by the ones used in [10, 9], and an example is presented in Example 1, about instance #1 of the TPP domain. Intuitively, the original action $drive$ can be executed even if the preference formula p-drive is not satisfied (and a related cost w is paid). The action $pdrive$ (resp. $dummy_{dr}$) takes into account if $drive$ was executed and the preference was (resp. not) satisfied. In the second case, a penalty has to be paid, and this is mimicked by adding a new dummy literal as effect (*goal-p-drive*, and the cost w is applied to its satisfaction).

Example 1. *The action $drive$ in the original instance:*

(:action drive
:parameters (?t - truck ?from ?to - place)
:precondition (and (at ?t ?from) (connected ?from ?to)
(preference p-drive (and
(ready-to-load goods1 ?from level0)
(ready-to-load goods2 ?from level0)
(ready-to-load goods3 ?from level0))))
:effect (and (not (at ?t ?from)) (at ?t ?to)))

is expressed with the new actions $pdrive$ and $dummy_{dr}$:

[4] We consider that at most one preference formula in expressed on the preconditions of an action: this is the case for all domains we consider in this paper. If this would not be the case, we should consider their power set.

(:action pdrive
:parameters (?t - truck ?from ?to - place)
:precondition (and (at ?t ?from) (connected ?from ?to)
(ready-to-load goods1 ?from level0)
(ready-to-load goods2 ?from level0)
(ready-to-load goods3 ?from level0))
:effect (and (not (at ?t ?from)) (at ?t ?to)))

(:action dummy$_{dr}$
:parameters (?t - truck ?from ?to - place)
:precondition (and (at ?t ?from) (connected ?from ?to)
(not (and (ready-to-load goods1 ?from level0)
(ready-to-load goods2 ?from level0)
(ready-to-load goods3 ?from level0))))
:effect (and (not (at ?t ?from)) (at ?t ?to) (goal-p-drive)))

About goal preferences and metric, Example 2 shows how we deal with them.

Example 2. *(Soft) Goals and the metric in the original instance are of the form:*

(:goal (and
(preference p4A (and (ready-to-load goods3 market1 level0)
(loaded goods3 truck1 level0)))
...
(preference p0A (stored goods3 level1))
...
))

(:metric minimize (+ (1 (is-violated p0A))*
...
(16 (is-violated p4A))*
(w (is-violated p-drive))))*

For each goal preference we introduce a (dummy) action whose precondition is the preference, and the effect is a (dummy) literal, e.g., for preference p4A

(:action dummy-p4A
:parameters ()
:precondition (and (ready-to-load goods3 market1 level0)
(loaded goods3 truck1 level0))
:effect (and (goal-p4A))).

At this point, no more "preference" construct is anymore in the problem. After this step, these new actions are compiled into (possibly multiple) STRIPS actions by using an existing tool (we have used ADL2STRIPS based on the LPG planner (see, e.g., [18, 19])[5]. The planning problem at hand defines a metric, i.e., a linear function with costs associated to the violation of goals and/or actions preconditions: in our formulation, such costs are now associated to the (un)satisfaction of the newly introduced dummy literals, i.e., a state in which such literals hold (or not).[6] If the cost is a real number, we have multiplied it by 10^n, where n is the maximum number of (significant) decimal digits in the problem. Specifically, the idea is to minimize the violation of preferences (expressed with (*is-violated p*) as in PDDL3.0, having the following meaning: given a preference p, *is-violated p*) takes value 1 if the preference is not satisfied, and 0 otherwise [7]). With our formulation, the new goal literals of introduced actions are reached when a preference is satisfied and this is "mimicked" by the related action's execution. Thus, the characterization of the metric function in Example 2 can be expressed with the following (linear) optimization function:

$$\textit{max: } +1 \times \pi\textit{(goal-p0A)} + \ldots + 16 \times \pi\textit{(goal-p4A)} - \sum_{i=1}^{n} +w \times \pi\textit{(goal-p-drive}_i\textit{)} \quad (1)$$

where π is a satisfying interpretation, and $\pi(p)$ is 1 if p is true, and 0 otherwise. Note that w=1 in tpp1. Eq. (1) considers the (simplified) setting in which only 1 STRIPS action has been created in place of a non-STRIPS action, whose name is the same as the original action.

At implementation level, we have modified SATPLAN at each makespan of the SATPLAN's approach, until the optimal. Thus, our compilation allows to find plans with optimal metrics at fixed makespan. Further, note that, while literals related to goal preferences can be implicitly considered to hold only "at the end" modality, i.e., at the final makespan, this is not the for the ones related to preconditions that can, in general, hold at any time stamps, unless we know that, instead, STRIPS actions can be only executed once (e.g., this is the case for real-world planning domain like blocks-world and logistics). In this second case the optimization function is:

$$\textit{max: } +1\ \pi\textit{(goal-p0A)} + \ldots + 16\ \pi\textit{(goal-p4A)} - +w\ \pi\textit{(goal-p-drive)} \quad (2)$$

The changes in SATPLAN were mainly related to the creation of formulas in Max-SAT and PB formats instead of the DIMACS format for SAT formulas in Conjunctive Normal Form (CNF), by adapting the plain CNF creation to the new formats. Weighted Partial Max-SAT problems require an update to the CNF format to include in the header line a positive integer number (top). top is a weight always

[5] `http://zeus.ing.unibs.it/lpg/`, i.e., the one used in the IPC-5.

[6] It is also possible to define the metric on actions with the related costs (if any), e.g., like [9]. In this paper, however, the definition is on states, similar to PDDL3.

greater than the sum of the weights of violated soft clauses. Then, each clause in the CNF format is modified by adding a positive integer number: this integer corresponds to top for hard clauses, while it is the related weight for a soft clause. The PB format, instead, requires the following changes: assuming k is the number of soft goals and actions preconditions, an optimization function of a form similar in structure to Eq. (1) and Eq. (2) (in PB-like syntax):

$$max: \ +w_1x_1 + w_2x_2 \cdots + w_kx_k \tag{3}$$

has to be explicitly added at the top of the instance, and then each SAT clause is expressed through the corresponding PB constraint. Assuming $w_k > 0$, the overall reduction is achieved with a set of new variables $x_1, x_2, \ldots, x_k$ involved in the optimization part (3), which are added as positive (resp. negative) literal to each soft goals and to each added effects of $dummy$ action introduced (e.g., $goal\text{-}p\text{-}drive$ in $dummy_{dr}$ action). Such new variables play the role of "preference selectors": if a goal is reached, the related selector is negated, thus the corresponding costs is not counted; on the contrary for each variables corresponding to actions preferences: if the effect is reached, this means that we have to pay a cost, because the action has been executed with unsatisfied soft preconditions. In Eq. (3), this corresponds to the related weight counted (negatively).

3 Experimental evaluation

We have used the best solvers that have participated to Max-SAT and PB evaluations along the years, with emphasis on the (more recent) "Weighted Partial" and "OPT-SMALL-INT" categories, the last being part of PB evaluations, and where (i) PB constraints correspond to SAT clauses; (ii) there is no constraint with a sum of coefficients greater than 2^{20} (20 bits), and (iii) the objective function is linear. Specifically, the solvers we have considered are: MINIMAXSAT ver. 1.0, based on MINISAT+ ver. 1.13, WMAXSATZ ver. 2.5, INCWMAXSATZ, MSUNCORE ver. 1.2 and 4; MINISAT+ ver. 1.14, GLPPB ver. 0.2, BSOLO ver. 3.0.17, SAT4J ver. 2.1 and SCIPSPX ver. 1.2.0.[7] MINIMAXSAT and SAT4J read instances in both partial weighted Max-SAT and PB formats, thus they are evaluated on both formulations. Regarding MSUNCORE, we only show results for ver. 1.2, given that the results for ver. 4.0 are very similar on the evaluated instances.

Given that the Weighted Partial Max-SAT and PB communities are mainly interested in instances with solutions, we focus on the results obtained by the various solvers on the first satisfiable Weighted Partial Max-SAT/PB instance we create,

[7] Solvers have been downloaded from `http://www.lsi.upc.edu/~fheras/docs/m.tar.gz`, `http://www.minisat.se/MiniSat+.html`, `http://www.eecs.umich.edu/~hsheini/pueblo`, `http://forge.ow2.org/projects/sat4j/`, `http://www.csi.ucd.ie/staff/jpms/soft/soft.php`, `http://scip.zib.de/`, or obtained by request to the authors. We have used the version submitted to the evaluations, or the last available.

which corresponds to extensions of the first satisfiable SAT formula following the SATPLAN approach. As we have already noted, in planning, such solutions would correspond to optimal, i.e., minimum, in plan metrics, plans at fixed, optimal makespan. Further, we consider the case where actions can be executed at most once. We show all instances that we could compile with the ADL2STRIPS tool, and then can be solved by at least one of the solvers. Some Pathways, TPP from #11 to #16, Trucks from #3 to #7, and Openstacks #1 (as numbered in the IPC-5) instances could be compiled but not solved by any system (for the instances of the last two domains, checking even satisfiability is hard for MINISAT). They thus provide challenging benchmarks for state-of-the-art Max-SAT and PB solvers. The timeout has been set to 900s on a Linux box equipped with a Pentium IV 3.2GHz processor and 1GB of RAM. In the tables, "TIME" means that the instance is not solved within the time limit. Results are presented in Figures 1-4 for the Pathways and TPP domains, given they are a consistent number, and in Tables 1 and 2 for the other domains. Figure 1 (resp. 2) contains the results for Max-SAT (resp. PB) and solvers on most of the Pathways instances: x-axis contains instances, as numbered in the IPC-5, while y-axis respective CPU times (in log scale). We preferred this way of presenting the results, instead of the one in the Max-SAT evaluation, because it maintains the correspondence between the instance and the related solvers performance. In Figures 1 and 2 results are mixed: while often one between INCWMAXSATZ and MINIMAXSAT is the best performing solver, SAT4J is the only system able to solve all the instances presented. In Figure 2, MINISAT+ is the best overall performing solver, followed by MINIMAXSAT and BSOLO. Figures 3 and 4[8] are structured as Figures 1 and 2. In Figure 3, on the largest instances MINIMAXSAT has superior performance than all other systems, followed by SAT4J. Results on small instances are different, and the other solvers perform better. In Figure 4, MINISAT+ shows the best performance followed by BSOLO and MINIMAXSAT.

Tables 1 and 2 contain results for the Storage and Trucks domains: the first column is the instance (again, as numbered in the IPC-5), and the other columns contain the results for the various systems. Distinguish performances are obtained by WMAXSATZ/INCWMAXSATZ and MINISAT+ in the Storage domain, and by MSUNCORE and MINIMAXSAT in the Trucks domain (even if only on 2 instances, the behavior seems to be clear).

Overall, MINISAT+ is the best performing system on the benchmarks analyzed, given it is the only system able to solve all the instances presented within the time limit, often in "short" time. This reminds and confirms results in the PB report [4], where MINISAT+ showed best performance on instances containing a (vast) majority of constraints corresponding to SAT clauses. Detailed observations are in the following. MINIMAXSAT and SAT4J, the solvers evaluated on both formulations, show different behaviors: the first is better on PB instances, the second is better on Max-SAT. About MINIMAXSAT, this is likely due to the fact that MINIMAXSAT

[8] GLPPB has not been considered here given it can not solve any instance.

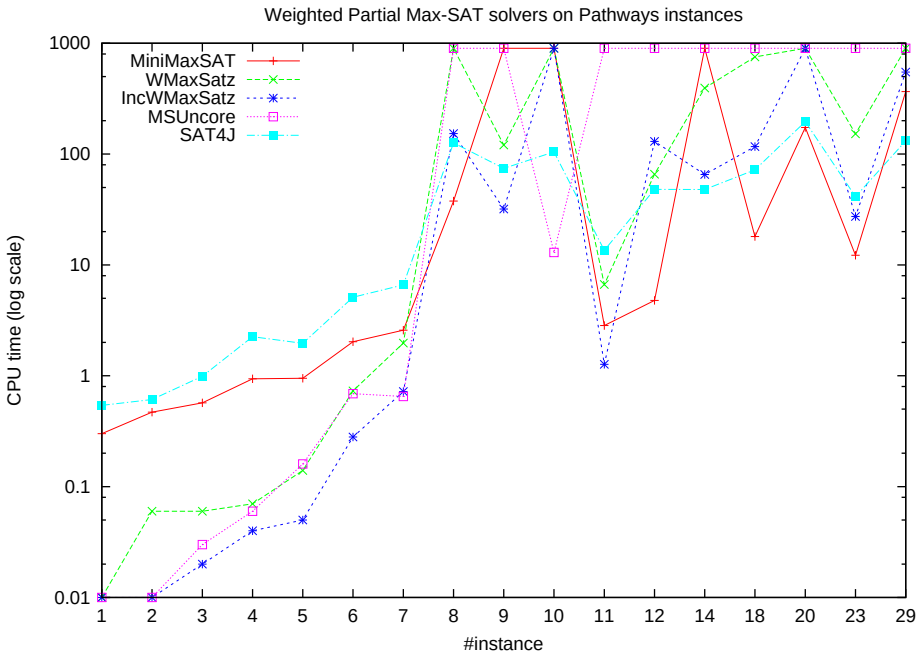

Figure 1: . Results of Max-SAT solvers on the Pathways domain.

instance	MINIMAXSAT	WMAXSATZ	INCWMAXSATZ	MSUNCORE	SAT4J
storage1	0.21	0.03	0.01	0.01	0.32
storage2	0.44	0.04	0.01	0.02	0.65
storage3	0.59	0.05	0.02	0.07	1.45
storage4	0.71	0.1	0.04	0.22	2.8
storage5	58.79	0.81	0.45	TIME	16.35
storage6	TIME	14.4	9.33	TIME	70.6
storage7	TIME	21.19	25.09	TIME	365.53
trucks1	7.7	TIME	TIME	0.74	359.17
trucks2	308.92	TIME	TIME	17.93	TIME

Table 1: Results of Max-SAT solvers on Storage and Trucks domains.

is based on an (early) version of MINISAT+. Pathways and Trucks domains share an observation: all unsolved instances have more than 10K variables and 200K clauses. The significant difference in the ratio r between sum of the weights of satisfied goals and top, which is much higher for Trucks, seems to be the reason for different solvers to perform best of the Pathways and Trucks domains. A last observation is devoted to the excellent results of WMAXSATZ and INCWMAXSATZ

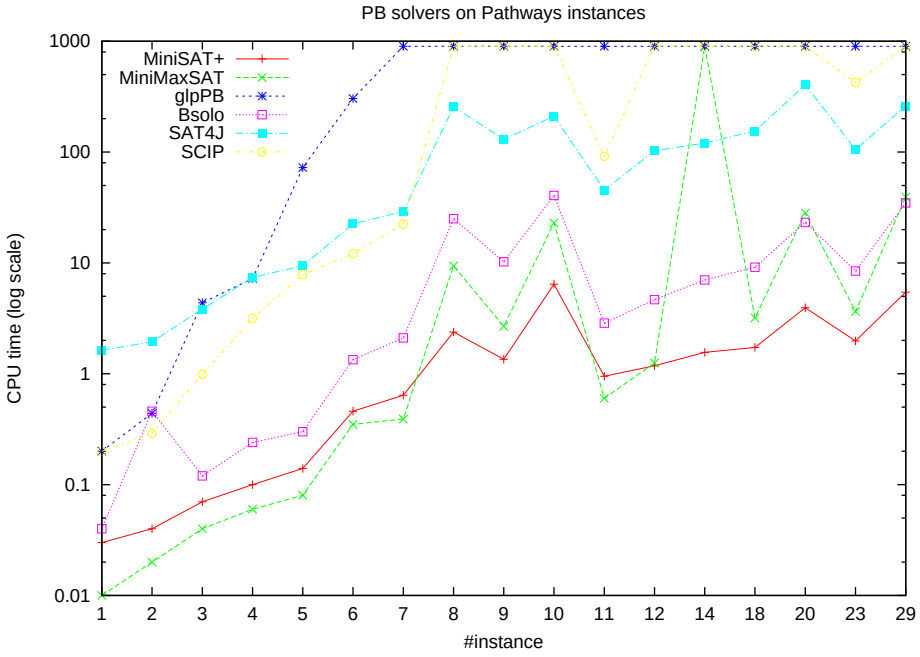

Figure 2: . Results of PB solvers on the Pathways domain.

instance	MINISAT+	MINIMAXSAT	GLPPB	BSOLO	SAT4J	SCIPSPX
storage1	0.01	0.01	0.03	0.03	0.65	0.06
storage2	0.02	0.02	1.14	0.32	1.91	0.2
storage3	0.08	0.03	TIME	0.46	4.15	0.38
storage4	0.21	0.09	TIME	0.97	9.7	0.9
storage5	2.47	2.87	TIME	62.76	48.18	4.38
storage6	4.99	7.83	TIME	TIME	151.35	140.52
storage7	36.08	TIME	TIME	TIME	678.74	51.66
trucks1	5.19	3.11	TIME	119.59	TIME	TIME
trucks2	385.07	54.84	TIME	TIME	TIME	TIME

Table 2: Results of PB solvers on Storage and Trucks domains.

on the Storage domain: large instances are characterized by a very high number of variables, clauses, soft goals and r. This is quite surprising, wrt the dimension of the instances, given they are "look-ahead" solvers. By inspection of their behaviors, the reason seems to be the very good bounds they provide initially.

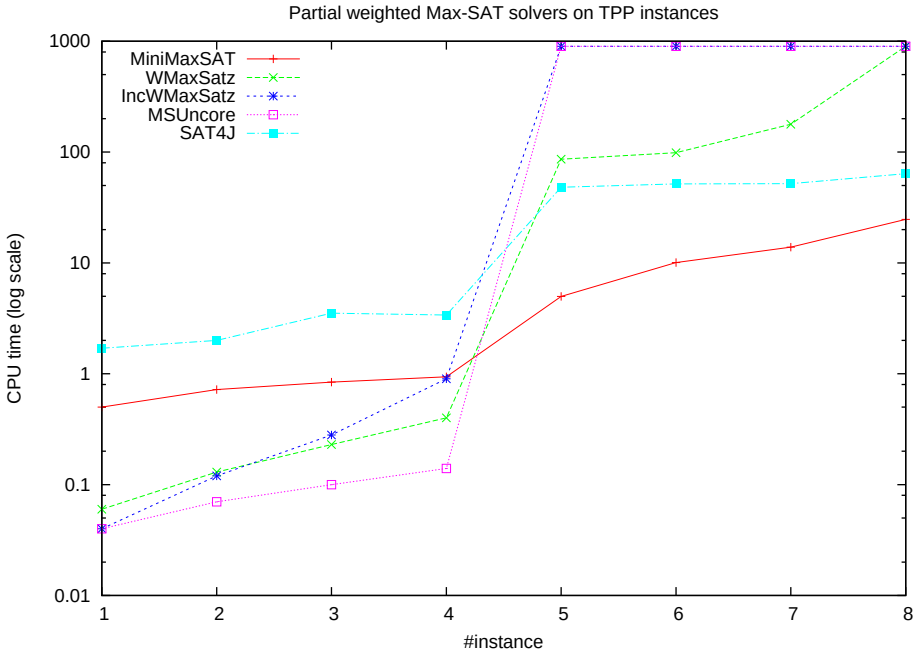

Figure 3: . Results of Max-SAT solvers on the TPP domain.

4 Conclusions

We have presented an experimental analysis on over-subscription planning problems from the IPC-5 with quantitative preferences on goals and/or actions preconditions, expressed (at fixed makespan) as Max-SAT and PB problems. The analysis reveals which, at the moment, is the best system on these benchmarks, and identifies challenging benchmarks that we plan to submit to the next evaluations. Future work includes the modeling and evaluation of instances from other IPC-5 and IPC-6 domains. The ultimate goal is to extend the planning as satisfiability framework, and SATPLAN, to effectively reason with planning with preferences, e.g., actions preconditions, soft goals and action costs, by means of a reduction to a series of Max-SAT/PB problems, and comparing the resulting satisfiability planner with state-of-the-art in the fields such as SGPLAN [20, 21] and GAMER [22]. A preliminary step towards the last point is presented in [23].

Acknowledgment. The author would like to thank Josep Argelich and Han Lin for providing, and getting support to, their solvers. Alessandro Saetti is, instead, thanked for his help with the IPC-5 benchmarks, and Menkes van den Briel for insights on the approach based on Integer Programming. Part of this work has been done while the author has been with the Università degli Studi e-Campus.

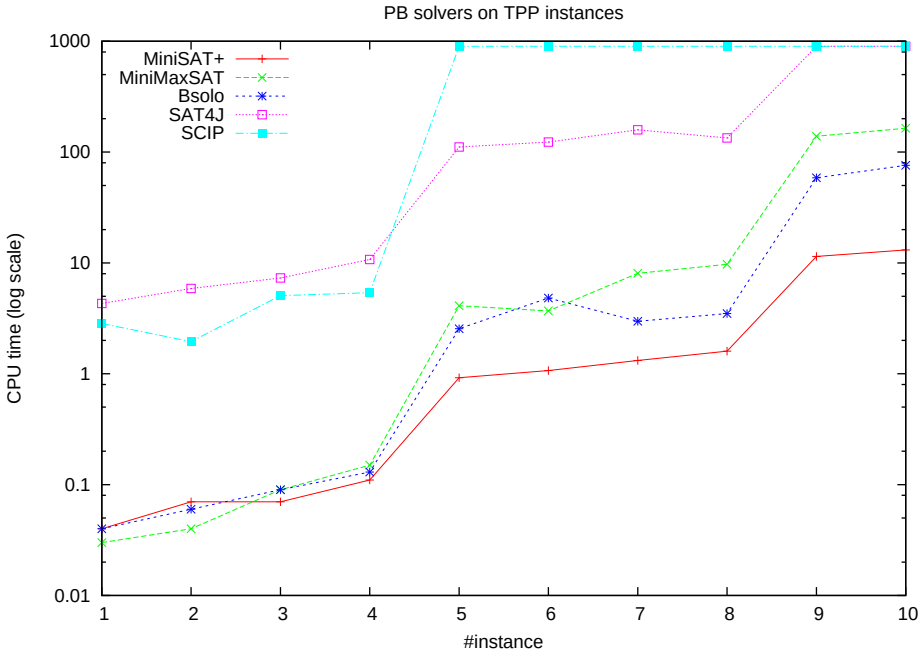

Figure 4: . Results of PB solvers on the TPP domain.

References

[1] Warners, J.P.: A linear-time transformation of linear inequalities into CNF. Information Processing Letters **68**(2) (1998) 63–69

[2] Bailleux, O., Boufkhad, Y.: Efficient CNF encoding of boolean cardinality constraints. In Rossi, F., ed.: Proc. of the 9th International Conference on Principles and Practice of Constraint Programming (CP 2003). Volume 2833 of Lecture Notes in Computer Science., Springer (2003) 108–122

[3] Argelich, J., Li, C.M., Manyà, F., Planes, J.: The first and second Max-SAT evaluations. Journal od Satisfiability, Boolean Modeling and Computation **4**(2-4) (2008) 251–278

[4] Manquinho, V.M., Roussel, O.: The first evaluation of pseudo-Boolean solvers (PB'05). Journal on Satisfiability, Boolean Modeling and Computation **2** (2006) 103–143

[5] van den Briel, M., Nigenda, R.S., Do, M.B., Kambhampati, S.: Effective approaches for partial satisfaction (over-subscription) planning. In McGuinness, D.L., Ferguson, G., eds.: Proc. of 19th National Conference on Artificial Intelligence (AAAI 2004), AAAI Press / The MIT Press (2004) 562–569

[6] Smith, D.E.: Choosing objectives in over-subscription planning. In Zilberstein, S., Koehler, J., Koenig, S., eds.: Proc. of 14th International Conference on Automated Planning and Scheduling (ICAPS 2004), AAAI (2004) 393–401

[7] Gerevini, A., Haslum, P., Long, D., Saetti, A., Dimopoulos, Y.: Deterministic planning in the 5th IPC: PDDL3 and experimental evaluation of the planners. Artificial Intelligence **173**(5-6) (2009) 619–668

[8] Fikes, R., Nilsson, N.J.: Strips: A new approach to the application of theorem proving to problem solving. Artificial Intelligence **2**(3-4) (1971) 189–208

[9] Benton, J., Kambhampati, S., Do, M.B.: YochanPS: PDDL3 simple preferences and partial satisfaction planning. 5th Internation Planning Competition Booklet, pages 23-25. Available at [24] (2006)

[10] Gazen, B.C., Knoblock, C.A.: Combining the expressivity of UCPOP with the efficiency of Graphplan. In Steel, S., Alami, R., eds.: Proc. of the 4th European Conference on Planning (ECP 1997): Recent Advances in AI Planning. Volume 1348 of Lecture Notes in Computer Science., Springer (1997) 221–233

[11] Hoffmann, J., Edelkamp, S., Thiébaux, S., Englert, R., dos S. Liporace, F., Trüg, S.: Engineering benchmarks for planning: the domains used in the deterministic part of IPC-4. Journal of Artificial Intelligence Research **26** (2006) 453–541

[12] Kautz, H., Selman, B.: Unifying SAT-based and graph-based planning. In Dean, T., ed.: Proc. of the 16th International Joint Conference on Artificial Intelligence (IJCAI 1999), Morgan-Kaufmann (1999) 318–325

[13] Kautz, H., Selman, B.: SATPLAN04: Planning as satisfiability. In: 5th Internation Planning Competition Booklet, pages 45-47. Available at [24]. (2006)

[14] Gerevini, A., Long, D.: Plan constraints and preferences in PDDL3. In: Proc. of the ICAPS-2006 Workshop on Preferences and Soft Constraints in Planning. (2006) 46–53

[15] van den Briel, M., Kambhampati, S., Vossen, T.: Planning with preferences and trajectory constraints through integer programming. In: Proc. of the ICAPS Workshop on Planning with Preferences and Soft Constraints. (2006) 19–22

[16] Eén, N., Sörensson, N.: Translating pseudo-Boolean constraints into SAT. Journal on Satisfiability, Boolean Modeling and Computation **2** (2006) 1–26

[17] Pednault, E.: ADL and the state-transition model of action. Journal of Logic and Computation **4** (1994) 467–512

[18] Gerevini, A., Serina, I.: Fast planning through greedy action graphs. In: Proc. of 16th National Conference on Artificial Intelligence (AAAI 1999). (1999) 503–510

[19] Gerevini, A., Saetti, A., Serina, I.: Planning through stochastic local search and temporal action graphs in LPG. Journal of Artificial Intelligence Research **20** (2003) 239–290

[20] Hsu, C.W., Wah, B.W., Huang, R., Chen, Y.: New features in SGPlan for handling preferences and constraints in PDDL3.0. In: 5th Internation Planning Competition Booklet, pages 39-41. Available at [24]. (2006)

[21] Hsu, C.W., Wah, B.W., Huang, R., Chen, Y.: Constraint partitioning for solving planning problems with trajectory constraints and goal preferences. In Veloso, M.M., ed.: Proc. of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007). (2007) 1924–1929

[22] Edelkamp, S., Kissmann, P.: Optimal symbolic planning with action costs and preferences. In Boutilier, C., ed.: Proc. of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009). (2009) 1690–1695

[23] Maratea, M.: Planning as satisfiability with ipc simple preferences and action costs. Technical report, Available at `http://www.star.dist.unige.it/~marco/Data/10constraints.pdf` (2010)

[24] `http://zeus.ing.unibs.it/ipc-5/booklet/i06-ipc-allpapers.pdf`: (2006)