An Assumption-Based Approach for Solving the Minimal S5-Satisfiability Problem

Jean-Marie Lagniez, Daniel Le Berre, Tiago de Lima, Valentin Montmirail

An Assumption-Based Approach for Solving The Minimal S5-Satisfiability Problem

Jean-Marie Lagniez[1], Daniel Le Berre[1], Tiago de Lima[1], and Valentin Montmirail[1]

CRIL, Artois University and CNRS, F62300 Lens, France
{lagniez,leberre,delima,montmirail}@cril.fr

Abstract. Recent work on the practical aspects on the modal logic S5 satisfiability problem showed that using a SAT-based approach outperforms other existing approaches. In this work, we go one step further and study the related minimal S5 satisfiability problem (MinS5-SAT), the problem of finding an S5 model, a Kripke structure, with the smallest number of worlds. Finding a small S5 model is crucial as soon as the model should be presented to a user, displayed on a screen for instance. SAT-based approaches tend to produce S5-models with a large number of worlds, thus the need to minimize them. That optimization problem can obviously be solved as a pseudo-Boolean optimization problem. We show in this paper that it is also equivalent to the extraction of a maximal satisfiable set (MSS). It can thus be solved using a standard pseudo-Boolean or MaxSAT solver, or a MSS-extractor. We show that a new incremental, SAT-based approach can be proposed by taking into account the equivalence relation between the possible worlds on S5 models. That specialized approach presented the best performance on our experiments conducted on a wide range of benchmarks from the modal logic community and a wide range of pseudo-Boolean and MaxSAT solvers. Our results demonstrate once again that domain knowledge is key to build efficient SAT-based tools.

Keywords: Modal Logic, S5, Incremental SAT, Minimisation

1 Introduction

Over the last twenty years, modal logics have been used in various areas of artificial intelligence like formal verification [1], database theory [2] and distributed computing [3] for example. More recently, the modal logic S5 was used for knowledge compilation [4] and in contingent planning [5]. Different solvers for different modal logics have been designed to decide the satisfiability of modal formulas since the 90's [6,7]. Some of them have been designed quite recently [8,9,10,11]. Despite the variety of techniques employed, none of them formally guarantees that, when a model is found, it is the smallest model possible (in number of worlds) for the input formula, in fact many of them do not even output a model but simply answer yes/no.

Providing a model, a certificate of satisfiability, is important to check the answer given by the solver. This is true both for the author of the solver or a user of that solver. This is mandatory nowadays in many solver competitions, among them the SAT competition [12]. It has also been shown that those models can help improving NP-oracle

based procedures [13]: a procedure requiring a polynomial number of calls to a yes/no oracle can be transformed into a procedure requiring only a logarithmic number of calls when the oracle can provide a model. Another example of the importance for an oracle to provide a model may be found in the model rotation technique [14], a method for the detection of clauses that are included in all MUSes (Minimal Unsatisfiable Sets) of a given formula via the analysis of models returned by a SAT oracle. Even if the theory does not guarantee a reduction of the number of oracle calls, in practice it provides a huge performance gain (up to a factor of 5) [15]. Finding the smallest model may be even more important in some contexts. The provided model usually has a meaning for the user, like in Hardware Verification [1] where the model is in fact an explanation of the bug found in the design of the hardware. The smaller the model, the more precise could be the location of the bug. It could also be the case that the model should be inspected by the user or displayed on a screen. Thus, the smaller, the better. There is a huge literature on minimizing models for SAT [16,17,18]. We are interested here in minimizing models for S5-SAT.

Our goal in this paper is to propose techniques to compute the smallest S5-model, in number of worlds, for a given input formula. We call this problem MinS5-SAT. We focus exclusively on the modal logic S5, for which the satisfiability problem is NP-complete [19] as for the classical propositional logic (CPL) [20]. We propose and compare different techniques. (1) The first obvious technique is based on a translation into CPL. The parameter given to the translation is the number n of possible worlds that the solution model is assumed to have. Linear or dichotomous search can be used to minimize the S5-model. (2) The second technique adds to the previous encoding selector-variables in order to activate or deactivate worlds. Finding a minimal S5-model in this case amounts to an equivalent MaxSAT problem [21], or, more surprisingly, to a MSS problem. Thus we can rely on off-the-shelves MaxSAT solvers or MSS-extractors to solve the original problem. (3) The last technique goes one step further from the two previous approaches. Thanks to a specific property of modal logic S5, we interpret the set of selectors causing the inconsistency of the formula to reach the theoretical upper bound faster. We compare these different techniques and show empirically which one better suits the benchmarks used. All benchmarks we could find for mono-agent S5 are randomly generated or created automatically following a pattern ("crafted"). (Note that reasoning about knowledge problems, such as those in [22], are all multi-agents.) However, we know from the SAT community [12] that the performance of a solver can be significantly different when the problem is randomly generated, when it is "crafted" or when it models a "real" problem which has some kind of "structure". Thus, we generated new S5 benchmarks translated from planning problems with incomplete information about the initial state (with sensing and full observability) [23] to complete the picture. We choose planning problems to obtain structured benchmarks requiering relatively large Kripke models.

In the reminder of this article, we first present the modal logic S5 and define the MinS5-SAT problem. Then, we provide a first approach to solve MinS5-SAT using a SAT-oracle and selector variables. We provide a translation from MinS5-SAT problems to equivalent MSS-extraction problems. We present a specific property of modal logic

S5 that speed up our initial SAT-based approach. Finally, we present the experimental results and conclude.

2 Preliminaries

2.1 Modal Logic S5

A central problem associated with any logic is the satisfiability problem, that is to decide whether a given formula has a model. The first complexity results for satisfiability in modal logic were achieved by Ladner [19]. He showed that the satisfiability problem in modal logic K (K-SAT) is in PSPACE and that the satisfiability problem in modal logic S5 (S5-SAT) is NP-complete. In this paper, we are interested in S5. In what follows, let $\mathbb{P}$ be a countably infinite non-empty set of propositional variables. The language $\mathcal{L}$ of modal logic is the set of formulas ϕ defined by the following grammar in BNF, where p ranges over $\mathbb{P}$:

$$\phi ::= \top \mid p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \mid \Box\phi \mid \Diamond\phi$$

The operators $\rightarrow$ and $\leftrightarrow$, defined by the usual abbreviations, are also used. A formula of the form $\Box\phi$ (*box phi*) means 'ϕ is necessarily true'. A formula of the form $\Diamond\phi$ (*diamond phi*) means 'ϕ is possibly true'.

Example 1. Let $\mathbb{P} = \{a, b\}$. $\phi = ((\Box\neg a \vee \Diamond b) \wedge \Diamond a \wedge \Box b)$ is a modal logic formula.

Formulas in $\mathcal{L}$ are interpreted using S5-structures [24], which are defined as follows:

Definition 1 (S5-Structure). *A S5-structure is a triplet $M = \langle W, R, \mathcal{I}\rangle$, where:*
W is a non-empty set of possible worlds;
R is a binary relation on W which is an equivalence relation ($\forall w.\forall v.\ (w, v) \in R$);
$\mathcal{I}$ is a function associating, to each $p \in \mathbb{P}$, the set of worlds from W where p is true.

Note that because R is an equivalence relation, we will omit it in the rest of this paper.

Definition 2 (Pointed S5-Structure). *A pointed S5-structure is a pair $\langle M, \omega\rangle$, where M is a S5-Structure and ω, called the actual world, is a possible world in W.*

In the remainder of this article, 'structure' means 'pointed S5-structure'. We define the size of a structure $\langle M, \omega\rangle$, noted $|M|$, as its number of worlds. Below, the satisfaction relation between such structures and formulas in $\mathcal{L}$ is defined.

Definition 3 (Satisfaction Relation). *Let $M = \langle W, R, \mathcal{I}\rangle$, an S5-structure. The satisfaction relation $\vDash$ between formulas and structures is recursively defined as follows:*

$$\langle M, \omega\rangle \vDash \top \quad \langle M, \omega\rangle \vDash p \text{ iff } \omega \in \mathcal{I}(p)$$
$$\langle M, \omega\rangle \vDash \neg\phi \text{ iff } \langle M, \omega\rangle \nvDash \phi$$
$$\langle M, \omega\rangle \vDash \phi \wedge \psi \text{ iff } \langle M, \omega\rangle \vDash \phi \text{ and } \langle M, \omega\rangle \vDash \psi$$
$$\langle M, \omega\rangle \vDash \phi \vee \psi \text{ iff } \langle M, \omega\rangle \vDash \phi \text{ or } \langle M, \omega\rangle \vDash \psi$$
$$\langle M, \omega\rangle \vDash \Box\phi \text{ iff for all } v \in W \text{ we have } \langle M, v\rangle \vDash \phi$$
$$\langle M, \omega\rangle \vDash \Diamond\phi \text{ iff there exists } v \in W \text{ such that } \langle M, v\rangle \vDash \phi$$

Definition 4 (Satisfiability). *A formula ϕ is satisfiable if and only if there exists a structure $\langle M, \omega \rangle$ that satisfies ϕ. Such a structure is called a 'model of ϕ'.*

Example 2. Here is a structure $\langle M, \omega_0 \rangle$ satisfying the formula ϕ from Example 1: $W = \{w_0, w_1, w_2\}$, $\mathcal{I} = \{\langle a, \{w_0\}\rangle, \langle b, \{w_0, w_1, w_2\}\rangle\}$. The size of $\langle M, \omega_0 \rangle$ equals 3.

As S5-SAT is NP-complete [19], we proposed a reduction from this problem to SAT in [10]. The reduction function takes as parameter the number of worlds n and is defined as follows:

Definition 5 (Translation function tr**).** *Let $\phi \in \mathcal{L}$.*

$$\mathrm{tr}(\phi, n) = \mathrm{tr}'(\phi, 1, n)$$

$$\mathrm{tr}'(p, i, n) = p_i \qquad \mathrm{tr}'(\neg\psi, i, n) = \neg\, \mathrm{tr}'(\psi, i, n)$$

$$\mathrm{tr}'(\psi \wedge \chi, i, n) = \mathrm{tr}'(\psi, i, n) \wedge \mathrm{tr}'(\chi, i, n) \qquad \mathrm{tr}'(\psi \vee \chi, i, n) = \mathrm{tr}'(\psi, i, n) \vee \mathrm{tr}'(\chi, i, n)$$

$$\mathrm{tr}'(\Box\psi, i, n) = \bigwedge_{j=1}^{n} (\mathrm{tr}'(\psi, j, n)) \qquad \mathrm{tr}'(\Diamond\psi, i, n) = \bigvee_{j=1}^{n} (\mathrm{tr}'(\psi, j, n))$$

The function 'tr' is satisfiability preserving if n is large enough. Moreover, some additional simplifications are performed to avoid outputting a very large formula (e.g., the Tseitin algorithm). See [10] for more details.

Example 3. Let ϕ the formula in Example 1. Its translation $\mathrm{tr}(\phi, 2)$ is $((\neg a_1 \vee b_1 \vee b_2) \wedge (\neg a_2 \vee b_1 \vee b_2) \wedge (a_1 \vee a_2) \wedge (b_1 \wedge b_2))$.

2.2 Unsatisfiable Cores

Recent SAT solvers are incremental, i.e., they are able to check the satisfiability of a formula "under assumptions" [25] and are able to output a core (a "reason" for the unsatisfiablity of the formula). The use of unsatisfiable cores is key to many applications, such as MaxSAT [21], MCS (Minimal Correction Set) [26], MUS (Minimal Unsatisfiable Set) [27]. The unsatisfiable core is defined as follows:

Definition 6 (Unsatisfiable Core under Assumptions). *Let Σ be a satisfiable CPL formula in CNF built using Boolean variables from $\mathbb{P}$. Let A be a consistent set of literals built using Boolean variables from $\mathbb{P}$ such that $(\Sigma \wedge \bigwedge_{a \in A} a)$ is unsatisfiable. $C \subseteq A$ is an unsatisfiable core (UNSAT core) of Σ under assumptions A if and only if $(\Sigma \wedge \bigwedge_{c \in C} c)$ is unsatisfiable.*

Definition 7 (SAT Solver under Assumptions). *Let Σ be a CPL formula in CNF. A SAT solver for Σ, given assumptions A, is a procedure which provides a pair $\langle r, s \rangle$ with $r \in \{\mathrm{SAT}, \mathrm{UNSAT}\}$ such that if $r = \mathrm{SAT}$ then s is a model of Σ, else if $r = \mathrm{UNSAT}$ then s is an UNSAT core of Σ under assumptions A.*

2.3 MSS and co-MSS

The problem of computing a Maximal Satisfiable Set of clauses (MSS problem) consists of extracting a maximal set of clauses from a formula in CNF that are consistent together [28]. The minimal correction subset (MCS or co-MSS) is the complement of its MSS.

Definition 8. *Let Σ be a given unsatisfiable formula in CNF. $S \subseteq \Sigma$ is a Maximal Satisfiable Subset (MSS) of Σ if and only if S is satisfiable and $\forall c \in \Sigma \setminus S$, $S \cup \{c\}$ is unsatisfiable.*

Definition 9. *Let an unsatisfiable formula Σ in CNF be given. $C \subseteq \Sigma$ is a Minimal Correction Subset (MCS or co-MSS) of Σ if and only if $\Sigma \setminus C$ is satisfiable and $\forall c \in C$, $\Sigma \setminus (C \setminus \{c\})$ is unsatisfiable.*

3 The MinS5-SAT problem

As pointed in [10], the necessary number of worlds to S5-satisfy a formula is bound by $\mathrm{dd}(\phi) + 1$, where $\mathrm{dd}(\phi)$ is given in Definition 10 below.

Definition 10 (Diamond-Degree). *The diamond degree of $\phi \in \mathcal{L}$, noted* $\mathrm{dd}(\phi)$*, is defined recursively, as follows:*

$$\begin{aligned}
\mathrm{dd}(\phi) &= \mathrm{dd}'(\mathrm{nnf}(\phi)) \\
\mathrm{dd}'(\top) &= \mathrm{dd}'(\neg\top) = \mathrm{dd}'(p) = \mathrm{dd}'(\neg p) = 0 \\
\mathrm{dd}'(\phi \wedge \psi) &= \mathrm{dd}'(\phi) + \mathrm{dd}'(\psi) \\
\mathrm{dd}'(\phi \vee \psi) &= \max(\mathrm{dd}'(\phi), \mathrm{dd}'(\psi)) \\
\mathrm{dd}'(\Box\phi) &= \mathrm{dd}'(\phi) \qquad \mathrm{dd}'(\Diamond\phi) = 1 + \mathrm{dd}'(\phi)
\end{aligned}$$

We denote by $\mathrm{nnf}(\phi)$ the formula ϕ in negation normal form (the negation applies only to propositional variables). Thus, we have that $\mathrm{tr}(\phi, \mathrm{dd}(\phi) + 1)$ is equisatisfiable to ϕ. Even if, in practice, we obtain very good results using this value as upper-bound, it seems far from the optimal value in all cases. For instance, in the model of Example 1, we can see redundancies: two worlds contain the same valuation. In contexts where the size of the returned model is critical, it makes sense to try to minimize it.

Consequently, in this article, we are interested in finding a model for a formula in S5 with the smallest number of worlds in its S5-structure. We call this problem the minimal S5 satisfiability problem and we define it as follows:

Definition 11 (Minimal S5 Satisfiability). *A formula ϕ is min-S5-satisfied by a structure $\langle M, \omega\rangle$ (noted $\langle M, \omega\rangle \models_{\min} \phi$) if and only if $\langle M, \omega\rangle \models \phi$ and ϕ has no model $\langle M', \omega'\rangle$ such that $|M'| < |M|$.*

Definition 12 (Minimal S5 Satisfiability Problem). *Let a formula ϕ in $\mathcal{L}$ be given. The minimal S5 satisfiability problem (MinS5-SAT) is the problem of finding a structure $\langle M, \omega\rangle$ such that $\langle M, \omega\rangle \models_{\min} \phi$.*

Let us remark that obtaining the minimal model for ϕ is not as simple as merging the worlds with the same valuations into only one world in any model of ϕ. The minimality cannot be guaranteed that way. Let us go back to the Example 2 to illustrate this. There, we have $\mathrm{dd}(\phi) + 1 = 3$. If we remove the redundancy, we obtain a model of size 2. However, ϕ is also satisfied by the following structure containing only one world: $W = \{w_0\}$, $\mathcal{I} = \{\langle a, \{w_0\}\rangle, \langle b, \{w_0\}\rangle\}$, which is a minimal model.

A very simple way to tackle this problem is to use the solver S52SAT [10] with a linear search strategy. Roughly, the procedure starts by trying structures of size $b = 1$. If no model is found, it iterates the process, each time increasing the value of b by 1. It iterates until a model of ϕ is found or the upper bound $\mathrm{dd}(\phi)+1$ is reached. This strategy is called 1toN. It is of course also possible to do it in reverse order: the procedure starts with $b = \mathrm{dd}(\phi) + 1$ and decreases the value of b by 1 (this is called Nto1). Yet another possibility is to use a binary search (called Dico).

However, these approaches are very naive. If we take, for instance, 1toN when the solution is a model of size m, it will perform m translations from S5 to SAT and then m calls to a SAT solver. The problem here is that such strategy does not take advantage of the previous UNSAT answer of the SAT solver to solve the new formula.

Modern SAT solvers are able to take advantage of previous calls when they are used incrementally [29]. The usual way to do that is to add selectors (assumptions) to the input formula and to get as output, on the suitable cases, some kind of "reason" for its unsatisfiability, in terms of these selectors. We propose here a way to add such selectors in the translation from S5 to SAT.

4 Preliminary step: an assumption-based translation

The translation 'tr' proposed in [10] is based on a simple, yet effective, idea: let ϕ be the input formula, every sub-formula of the form $\Box\psi$ is translated to $\bigwedge_{i=1}^{n} \mathrm{tr}(\psi, i, n)$, whereas sub-formulas of the form $\Diamond\psi$ are translated to $\bigvee_{i=1}^{n} \mathrm{tr}(\psi, i, n)$. The number n is the number of possible worlds of the model being constructed. If we set $n = \mathrm{dd}(\phi) + 1$, we are guaranteed to have an equi-satisfiable formula on the output.

In order to take advantage of the ability of modern SAT solvers to return a reason for unsatisfiability, we can add selector variables s_i to enable or disable worlds w_i. We update the translation of $\Box\psi$ to $\bigwedge_{i=1}^{n}(\neg s_i \vee \mathrm{tr}(\psi, i, n))$ and the one of $\Diamond\psi$ to $\bigvee_{i=1}^{n}(s_i \wedge \mathrm{tr}(\psi, i, n))$. Worth noticing that due to the simplifications authorized in S5, the modalities cannot be embedded modalities. The modal depth equals 1. While the resulting CNF is be bigger than the original one, the size of the S5-model will now be decided by the number of satisfied selector variables. The complete translation function is given below:

Definition 13 (Translation with selectors). *Let $\phi \in \mathcal{L}$.*

$$\mathrm{tr_s}(\phi, n) = \mathrm{tr_s}'(\phi, 1, n)$$

$$\mathrm{tr_s}'(p, i, n) = p_i \qquad \mathrm{tr_s}'(\neg\psi, i, n) = \neg\,\mathrm{tr_s}'(\psi, i, n)$$

$$\mathrm{tr_s}'(\psi \wedge \delta, i, n) = \mathrm{tr_s}'(\psi, i, n) \wedge \mathrm{tr_s}'(\delta, i, n) \qquad \mathrm{tr_s}'(\psi \vee \delta, i, n) = \mathrm{tr_s}'(\psi, i, n) \vee \mathrm{tr_s}'(\delta, i, n)$$

$$\mathrm{tr_s}'(\Box\psi, i, n) = \bigwedge_{j=1}^{n}(\neg s_j \vee (\mathrm{tr_s}'(\psi, j, n)) \qquad \mathrm{tr_s}'(\Diamond\psi, i, n) = \bigvee_{j=1}^{n}(s_j \wedge (\mathrm{tr_s}'(\psi, j, n))$$

An S5-model has to have at least one possible world (the current world). W.l.o.g., we consider that the current world is the world number 1, thus s_1 is always set to true. In the remainder of this article, we denote by $\mathrm{tr_s}(\phi)$ the formula $(\mathrm{tr_s}(\phi, \mathrm{dd}(\phi)+1) \wedge s_1)$ and the set of all selectors of $\mathrm{tr_s}(\phi)$ is denoted by $\mathcal{S}(\phi)$ (i.e., $\mathcal{S}(\phi) = \{s_i \mid 1 \leq i \leq \mathrm{dd}(\phi)+1\}$).

Example 4 (Example of '$\mathrm{tr_s}$'). Let us go back to Example 1 and reuse the formula $\phi = ((\Box\neg a \vee \Diamond b) \wedge \Diamond a \wedge \Box b)$. Its translation $\mathrm{tr_s}(\phi, 3)$ is:

$$(\neg s_1 \vee (\neg a_1 \vee (s_1 \wedge b_1) \vee (s_2 \wedge b_2) \vee (s_3 \wedge b_3))) \wedge$$
$$(\neg s_2 \vee (\neg a_2 \vee (s_1 \wedge b_1) \vee (s_2 \wedge b_2) \vee (s_3 \wedge b_3))) \wedge$$
$$(\neg s_3 \vee (\neg a_3 \vee (s_1 \wedge b_1) \vee (s_2 \wedge b_2) \vee (s_3 \wedge b_3))) \wedge$$
$$((s_1 \wedge a_1) \vee (s_2 \wedge a_2) \vee (s_3 \wedge a_3)) \wedge ((\neg s_1 \vee b_1) \wedge (\neg s_2 \vee b_2) \wedge (\neg s_3 \vee b_3)) \wedge s_1$$

Intuitively, every formula with subscript i is a formula that is true at the possible world i. If the selector s_i is false, then the world i is not present in the model (and we do not care about the valuation of the propositions there in). Below, a formula that is equivalent to $\mathrm{tr_s}(\phi, 3)$ but with s_1 and s_2 activated and s_3 deactivated.

$$(\neg\top \vee (\neg a_1 \vee (\top \wedge b_1) \vee (\top \wedge b_2) \vee (\bot \wedge b_3))) \wedge$$
$$(\neg\top \vee (\neg a_2 \vee (\top \wedge b_1) \vee (\top \wedge b_2) \vee (\bot \wedge b_3))) \wedge$$
$$(\neg\bot \vee (\neg a_3 \vee (\top \wedge b_1) \vee (\top \wedge b_2) \vee (\bot \wedge b_3))) \wedge$$
$$((\top \wedge a_1) \vee (\top \wedge a_2) \vee (\bot \wedge a_3)) \wedge ((\neg\top \vee b_1) \wedge (\neg\top \vee b_2) \wedge (\neg\bot \vee b_3)) \wedge \top$$

This formula is equivalent to $(\neg a_1 \vee b_1 \vee b_2) \wedge (\neg a_2 \vee b_1 \vee b_2) \wedge (a_1 \vee a_2) \wedge (b_1 \wedge b_2)$, which is the same as the one presented in Example 3. It corresponds to the problem of deciding if ϕ is satisfiable in a model with 2 worlds.

As we can see, the problem of solving the minimal S5 satisfiability problem is now equivalent to the problem of satisfying $\mathrm{tr_s}(\phi, n)$ and minimize the number of s_i, for $i > 1$, assigned to true (or, equivalently, maximize the number of s_i assigned to false). Obviously, it can be seen as a pseudo-Boolean optimization problem [30], where the optimization function to be minimized is the number of selectors assigned to true. This problem is also often solved nowadays as an instance of the Partial MaxSAT problem [21], which consists in satisfying all the hard clauses (clauses that MUST be satisfied) and the maximum number of soft clauses (the clauses that are not mandatory). In our case, the hard clauses are those generated by the translation function, and the soft-clauses are the unit clauses $\{\neg s_i \mid s_i \in \mathcal{S}(\phi) \text{ and } i > 1\}$ built from the selector variables.

We can thus use state-of-the-art Partial MaxSAT solvers. However, it is not the only way, as we show in the following section. By considering the structure of S5-models, extracting a MSS can also be used to decide the problem.

W.l.o.g., in the following sections, we represent a set of unit soft clauses as the set of selectors composing it (eg.: $\{s_2, s_3, s_4\}$ rather than $\{\neg s_2, \neg s_3, \neg s_4\}$).

5 First insight: cardinality optimality equals subset optimality

In the Section 2.3, we defined the MSS problem. It is also possible to define a partial version of the MSS problem, where the objective is to compute a MSS such that some

given subset of the clauses (the hard clauses) must be satisfied. This problem is related to the Partial MaxSAT problem. In fact, a solution to a Partial MaxSAT problem is one of the biggest MSS that satisfies the set of hard-clauses. In general, a partial MSS is not a solution to a Partial MaxSAT problem but, in the specific case of MinS5-SAT, a partial MSS is also a solution to its corresponding Partial MaxSAT problem, which means that in that specific context, subset optimality (MSS) is equivalent to cardinality optimality (MaxSAT).

Proposition 1. *Let* $\Sigma = \mathrm{tr}_s(\phi, \mathrm{dd}(\phi) + 1)$ *a CNF, and let* χ *be the formula* $\bigwedge_{i=2}^{\mathrm{dd}(\phi)+1} \neg s_i$. *An MSS of* $(\Sigma \wedge \chi)$*, where* Σ *is the set of hard clauses, is also a solution to the Partial* MaxSAT *problem* $(\Sigma \wedge \chi)$.

The proof of Prop. 1 uses the following lemma.

Lemma 1. *Let* $\Sigma = \mathrm{tr}_s(\phi, n)$*, and let* χ *be the formula* $\bigwedge_{s_i \in \mathcal{S}'} \neg s_i$*, where* $\mathcal{S}' \subseteq \mathcal{S}(\phi)$*. If* $(\Sigma \wedge \chi)$ *is satisfiable then so is the formula* $(\Sigma \wedge \chi')$*, where* χ' *is obtained from* χ *by replacing the occurrences of one selector* $s \in \mathcal{S}'$ *by another selector* $s' \in \mathcal{S}(\phi) \setminus \mathcal{S}'$.

Proof (sketch). The proof is done by an induction on the length of the formula ϕ. In the induction base, $\Sigma = p$, for some $p \in \mathbb{P}$. We have $\Sigma = p_1$, $\chi = \neg s_1$ and $\mathcal{S}(\phi) = \{s_1\}$, which means that the claim is true (because $\mathcal{S}(\phi) \setminus \mathcal{S}' = \emptyset$). We have several cases on the induction step. Since their proofs are all similar, we show only one them here. Let $\phi = \Box\phi'$. We have $\Sigma = \bigwedge_{i=1}^{n}(\neg s_i \vee \mathrm{tr}_s(\phi', i, n))$, $\chi = \bigwedge_{s_i \in S'} \neg s_i$, and $\mathcal{S}(\phi) = \{s_1, \ldots, s_n\}$. Now, let χ' be obtained from χ where s_i is replaced by $s_j \in \mathcal{S}(\phi) \setminus S'$. If $(\Sigma \wedge \chi)$ is satisfied by a model M then we construct a new model M', which equals M except that the truth assignment of all propositional variables with subscript i are the same as those with subscript j. We immediately have that if $M \models \chi$ then $M' \models \chi'$. We also have that if $M \models \neg s_i$ then $M' \models \neg s_j$. Finally, for each $1 \leq i \leq n$, if $M \models \mathrm{tr}_s(\phi', i, n))$ then $M' \models \mathrm{tr}_s(\phi', j, n))$, by the induction hypothesis (since the length of ϕ' is strictly smaller than that of ϕ). Therefore, $M' \models \Sigma \wedge \chi'$.

Proof (of Prop. 1). Towards a contradiction, assume that there exists a MSS $\delta_1 = (\Sigma \wedge \chi_1)$, where $\chi_1 = \bigwedge_{s \in S_1} \neg s$, which is not the biggest one. Thus, there exists another MSS $\delta_2 = (\Sigma \wedge \chi_2)$, where $\chi_2 = \bigwedge_{s \in S_2} \neg s$ and such that $|S_1| < |S_2|$. Now, let $S_3 = S_2 \setminus \{s\} \cup \{s'\}$, where $s \in S_2$ and $s' \in S_1$. By Lemma 1, the formula $\delta_3 = (\Sigma \wedge \chi_3)$, where $\chi_3 = \bigwedge_{s \in S_3} \neg s$ is satisfiable, because it is δ_2 with one of the selectors of S_2 in χ_2 replaced by another selector. It is easy to see that one can keep replacing selectors in this set until we have the set S_k, such that $S_1 \subseteq S_k$. The formula $\delta_k = (\Sigma \wedge \chi_k)$, where $\chi_k = \bigwedge_{s \in S_k} \neg s$, is satisfiable, by applying Lemma 1 $|S_1|$ times. Then δ_k a MSS that includes δ_1, which contradicts the assumption. This means that every MSS of the initial formula is one of the biggest ones. Therefore, any MSS of $(\Sigma \wedge \chi)$ is also a solution to the partial MaxSAT problem $(\Sigma \wedge \chi)$. □

As a direct consequence of Prop. 1, we can always find a MSS such that the indexes of the selectors inside it are contiguous. This means that we can consider an optimisation that reduces the search space (breaks the symmetries), by adding the following:

$$\left(\bigwedge_{i=1}^{n-1} \neg s_i \rightarrow \neg s_{i+1}\right) \tag{1}$$

By giving as input $\mathrm{tr_s}(\phi, n)$ plus $\mathcal{S}(\phi)$, we can solve the MinS5-SAT problem with a MaxSAT solver, or a Pseudo Boolean (PB) solver. If we also add Equation 1 to the input, we can then use a MSS-extractor. However, we demonstrate in the following section that we can push the envelope by considering a dedicated approach using an incremental SAT solver with unsatisfiable cores.

6 Second insight: only core size matters

Consider the following example: let ϕ be the input formula and let $\mathrm{dd}(\phi) + 1 = 10$. We translate ϕ using selectors and start looking for a model for it. Assume that, after some computation, we conclude that 4 worlds cannot be deactivated altogether, i.e., if the selectors s_i, s_j, s_k and s_l are set to false, we have an inconsistency. We can infer from that information that we will need at least 7 worlds in the S5-model for ϕ. This comes from the fact that the '4 worlds which cannot be deactivated altogether' can be, in fact, any group of 4 worlds. Indeed, in the sequel, we demonstrate that if we have a group of m selectors forming an unsatisfiable core and the upper-bound equals n, then we need at least $(n - m + 1)$ worlds in the S5-model of the input formula.

Proposition 2. *Let $\phi \in \mathcal{L}$ such that $\mathrm{dd}(\phi) + 1 = n$. If C is an UNSAT core of ϕ under assumptions $\mathcal{S}(\phi)$ then $\mathrm{tr_s}(\phi, n')$ is unsatisfiable for all $n' \in \{1, \ldots, (n - |C|)\}$.*

Lemma 2. *If C is an UNSAT core of ϕ with assumptions $\mathcal{S}(\phi)$ then any set of literals $C' = \{\neg s \mid s \in \mathcal{S}(\phi)\}$ such that $|C'| = |C|$ is an UNSAT core of ϕ.*

Proof. Assume that C is an UNSAT core of ϕ with assumptions $\mathcal{S}(\phi)$. We have that $(\phi \wedge \bigwedge_{l \in C} l)$ is unsatisfiable. Now, towards a contradiction, also assume that there exists a set $C' = \{\neg s \mid s \in \mathcal{S}(\phi)\}$ such that $|C'| = |C|$ and $(\phi \wedge \bigwedge_{s \in C} \neg s)$ is satisfiable. By Lemma 1, we can obtain a new set D from C' by replacing the selectors in C' by those in C such that $(\phi \wedge \bigwedge_{s \in D} \neg s)$ is satisfiable. Because $D = C$, we have a contradiction. Therefore, any set of literals C' obtained as such is an UNSAT core of ϕ. □

Proof (of Prop. 2). The formula has n worlds. The SAT solver returns a core C of size m. So one of the selector has to be true. But due to Lemma 2, we have to put at least one selector to true to all the possible unsatisfiable cores of size m. Said otherwise, we must have $(n - m + 1)$ selectors to be true together, or the formula will be necessarily unsatisfiable. This also means that $\forall b' \in [1 \ldots (n - m)]$ $\mathrm{tr_s}(\phi, b')$ is unsatisfiable. □

Using this property, it is possible to construct an iterative algorithm which is based on incremental SAT. The SAT solver will be able to return an unsatisfiable core, and by interpreting it in the specific case of S5 as explained in Prop. 2, we can refine the bound used in the translation. The procedure starts by trying structures of size $b = 1$. If no model is found, it iterates the process, each time increasing the value of b by $(\mathrm{dd}(\phi) + 1 - |s| + 1)$ (where $|s|$ is the size of the core). It iterates until a model of ϕ is found or the upper bound $\mathrm{dd}(\phi) + 1$ is reached. Note that $|s|$ strictly decreases at each step, because we strictly increase the number of satisfied selectors. The approach $Dicho_c$ is similar.

7 Experimental results

We compared several different approaches to the MinS5-SAT problem: S52SAT [10] with five different strategies: 1toN$_c$, 1toN, Nto1, Dicho$_c$, Dicho. CNF plus MaxSAT solver: maxHS-b [31], mscg2015b [32], and MSUnCore [33]. Pseudo-Boolean (PB) translation plus *PB* solver: NaPS [34], SAT4J-PB [35], SCIP [36]. CNF plus symmetry breaking plus MCS extraction with the LBX solver [37].

To see the impact of our minimisation, we use the state-of-the-art modal logic S5 solver S52SAT with glucose (4.0) as embedded SAT solver [29] (with its caching activated). We selected MaxSAT solver which have shown good performances in the MaxSAT competition 2016 [38]. We also considered LMHS-2016 [39] but, unfortunately, we did not manage to compile it due its multiple links to other software and our configuration environment.

Despite our through research, we could not find benchmarks for modal logic S5. Due to this fact, we chose to use the following benchmarks for modal logics K, KT and S4: TANCS-2000 modalised random QBF (MQBF) formulae [40] complemented by additional MQBF formulae provided by [41]; LWB K, KT and S4 formulae [42], with 56 formulae chosen from each of the 18 parametrized classes, generated from the script given by the authors of [9]; and Randomly generated $3CNF_{KSP}$ formulae [43] of modal depths 1 and 2. The benchmarks are classified as SAT or UNSAT in [9,42]. However, we kept only the benchmarks satisfied in their original logic to see the impact of a potential use of a S5-solving as a preprocessor for other modal logics. We have no interest with the UNSAT benchmarks because the unsatisfiability in K,KT and S4 implies the unsatisfiability in S5. We also proposed new benchmarks based on planning with uncertainties in the intial states, to check the performance of the different approaches on structured benchmarks. In such planning problems, some fluent f may be initially true, initially false, or neither. I the latter case, two different initial situations are possible. As a result, instead of a single initial state s_0, we may have several different initial states, which are consistent with available knowledge about the system (see [23] for more details and applications). By construction, all instances considered here have a plan to minimize. Modal logic S5 formulas are generated with a CEGAR approach [44]. We increase the value of the bounded-horizon until we reach the smallest value for which there exists a plan as explained in [45].

We performed experimental evaluations on a variety of planning benchmarks. It includes the traditional conformant benchmarks, namely: Bomb-in-the-toilet, Ring, Cube, Omelet and Safe (see [46] for more details) modeled here as planning with uncertainties in the initial state. We also performed evaluations on classical benchmarks: Blocksworld, Logistics, and Grid, in which the authors of [47] introduced uncertainty about the initial state. All the benchmarks are available for download[1].

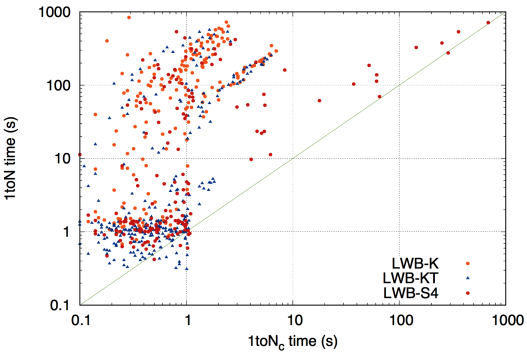
To select the "minimalizable" benchmarks, we set a time-out of 1500 seconds. We managed to solve 28 benchmarks out of the 119 available. We tried other solvers: Spartacus [48] solved 15 instances and SPASS [49] solved 5, with both being a subset of the 28 solved by S52SAT. Our generator has negligible execution times and is available for

[1] `https://fai.cs.uni-saarland.de/hoffmann/ff/cff-tests.tgz`

Table 1: #Instances solved in LWB

Method	**K**	**KT**	**S4**	**Total**
# Benchs	(185)	(279)	(160)	(624)
1toN	**185**	**279**	**160**	**624**
Nto1	17	34	2	53
Dicho	119	175	78	372
1toN$_c$	**185**	**279**	**160**	**624**
Dicho$_c$	135	201	100	436
maxHS	17	25	72	114
MSCG	74	65	103	242
MSUnCore	19	30	80	129
NaPS	126	64	71	261
SAT4J	18	27	58	103
SCIP	104	158	112	374
LBX	118	173	92	383
VBS	185	279	160	624

Fig. 1: Scatter-plot of 1toN vs 1toN$_c$



download[2]. Each of these benchmarks has a plan of size N (where N can be different for each benchmark) which has been verified. We then generated modal logic benchmarks from these instances by fixing the horizon at N, $N+1$, ..., $N+9$ having thus 280 benchmarks, all S5-satisfiable, to test our minimisation techniques.

The benchmarks and the different solvers (especially S52SAT, which is the one translating the formulas into propositional logic) are available[3]. The experiments ran on a cluster of Xeon 4 cores, 3.3 GHz, running CentOS 6.4. The memory limit is set to 32GB and the runtime limit is set to 900 seconds per solver per benchmark. In the following tables, we provide the number of benchmarks for which a minimal S5 model is found. In bold face, the best result for each row/column. The VBS (Virtual Best Solver) represents the union of the benchmarks solved by all the approaches.

7.1 State-of-the-art modal logics benchmarks

Logic WorkBench (LWB) Benchmarks All the results are reported in the Table 1. The difference in the results between the approaches using S52SAT and the MaxSAT solvers came from the fact that MaxSAT solvers cannot take into account inherent properties of modal logic S5. They have embedded cardinality constraints used to count the number of satisfied/falsified clauses to return the smallest model. By comparing the results of 1toN and 1toN$_c$ in number of benchmarks solved, one could think that selectors do not make much difference. But the runtime provides a different picture, as in the scatter plot depicted in Figure 1. The x-axis corresponds to the time used by 1toN$_c$ while the y-axis corresponds to the time used by 1toN to solve these problems. As expected, 1toN$_c$ performs less iterations and thus calls the SAT solver fewer times. We remark that the

[2] `http://www.cril.fr/~montmirail/planning-to-s5/`
[3] `http://www.cril.fr/~montmirail/s52SAT`

Table 2: #instances solved $3CNF_{KSP}$

Benchs	1toN	Nto1	Dicho	1toN$_c$	Dicho$_c$	maxHS	MSCG	MSUnCore	NaPS	SAT4J	SCIP	LBX	VBS
md=1 (62)	55	0	26	**62**	40	40	30	38	42	35	42	47	62
md=2 (27)	17	0	9	**27**	17	12	12	12	17	12	20	17	27
Total (89)	72	0	35	**89**	57	52	42	50	59	47	62	64	89

Table 3: #instances solved in $MQBF$

Benchs	1toN	Nto1	Dicho	1toN$_c$	Dicho$_c$	maxHS	MSCG	MSUnCore	NaPS	SAT4J	SCIP	LBX	VBS
qbf (56)	**56**	55	**56**	**56**	**56**	**56**	**56**	**56**	55	48	**56**	**56**	56
qbfS (171)	**171**	0	**171**	**171**	**171**	0	156	0	144	140	155	167	171
Total (227)	**227**	55	**227**	**227**	**227**	56	212	56	199	188	211	223	227

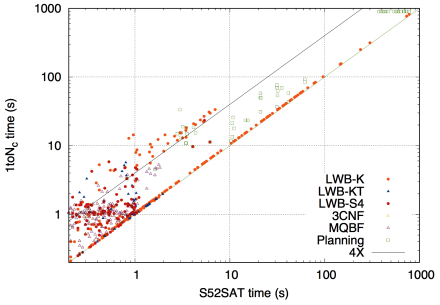
solver took less than 10 seconds for the majority of the instances. It turns out that it makes sense to consider this approach as a pre-processing for a more generic minimal modal logic SAT solver (eg., for logics K, KT and S4). Indeed if we find a minimal model in S5, we obtain in the same way an upper-bound on the size of the minimal model in K, KT and S4.

$3CNF_{KSP}$ ***benchmarks*** The randomly generated $3CNF_{KSP}$ formulae [43] of depths 1 and 2 consist of 1000 formulae, where 457 are satisfiable in modal logic K and 89 are satisfiable in S5. All the results are reported in the Table 2. As for LWB, 1toN and Dicho are better than Nto1 because the minimal models found are relatively small. It is interesting to notice that the modal depth of the formulas influences the result. This is surprising due to the fact that, in S5, all formulae can be reduced to modal depth 1. In fact, many instances with modal depth 2, that are SAT in modal logic K, are UNSAT in modal logic S5.

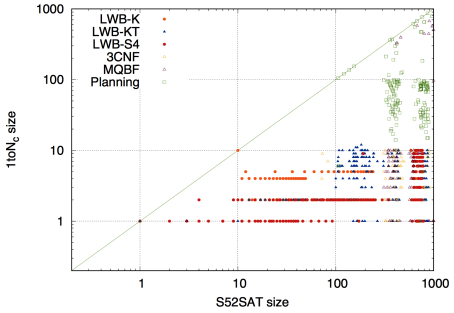
Randomly Modalized QBF (MQBF) benchmarks Originally, this benchmark set contains 1016 formulas, among them 617 are SAT while 399 are UNSAT in K. All the results are reported in the Table 3. Dicho, Dicho$_c$, 1toN and 1toN$_c$ approaches are better than the other ones. Moreover, it is interesting to see that the whole **qbf** family is in fact $S5$-satisfiable, even though they are normally used to evaluate modal logic K solvers. It is worth noticing that the performance of a MaxSAT or a PB approach are globally worse than the MSS-extaction approach. However, if we add the symmetry breaking from Equation 1 then the performances become equivalent.

Structured Benchmarks: Planning with uncertainties As in the random and crafted benchmarks before, we can see in Figure. 2c that the use of selectors allows us to solve more benchmarks. But, surprisingly, here the best approach is to use a dichotomic search instead of a linear search from 1 to N. This is mainly due to the size of the smallest model, which is rarely a small number, as it was the case in LWB for example. Moreover, each call to the SAT solver is more time-consuming because the instances are harder to solve in practice. This again reminds us that the benchmarks considered can influence the result obtained.

(a) S52SAT vs S52SAT-1toN$_c$ (time)



(b) S52SAT vs S52SAT-1toN$_c$ (size)



Method # Benchs	**block** (20)	**bomb** (30)	**cube** (100)	**omelet** (30)	**ring** (40)	**safe** (60)	**Total** 280
1toN	10	25	**100**	0	**40**	0	175
Nto1	5	10	20	0	20	0	55
Dicho	**20**	20	70	0	**40**	0	150
1toN$_c$	**20**	25	**100**	10	**40**	30	225
Dicho$_c$	**20**	**30**	**100**	**18**	**40**	**34**	**242**
maxHS	17	22	82	0	**40**	5	166
MSCG	**20**	25	72	0	**40**	4	161
MSUnCore	10	22	68	0	38	0	138
NaPS	**20**	**30**	74	2	**40**	8	174
SAT4J	**20**	20	58	0	33	0	131
SCIP	**20**	**30**	**100**	5	**40**	10	200
LBX	12	26	79	0	**40**	0	157
VBS	20	30	100	18	40	34	242

(c) #instances solved in Planning

(d) Spartacus vs S52SAT-1toN$_c$ (size)

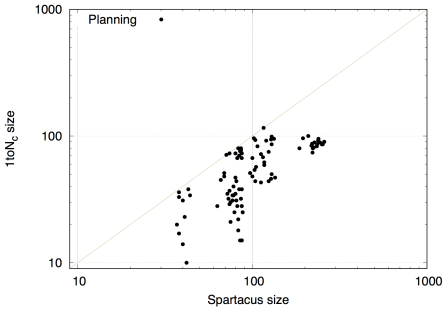


Fig. 2: Results on Planning and analysis of the overhead

Minimization overhead We can see on Figure. 2a that it requires only an acceptable over-head computation to get the smallest model possible instead of the first one returned by the solver (from less than 10s to less than 40s). On the other hand, as we can see on Figure. 2b the minimization can reduce drastically the size of the returned model. Moreover, we can also see the structural difference between randomly generated instances, that can finally be solved with only few worlds, and 'real-world' applications instances that need a larger number of worlds to be solved. There is also a gain against other solvers able to output a model (Figure 2d), such as Spartacus [48]. Note that we could only compare Spartacus models on planning problems because Spartacus is dedicated to modal logics K, KT and S4, not S5. However, on those specific benchmarks, since the modal depth is one and all K-models on those benchmarks are S5 models, we can compare their size. Note also that Spartacus outputs an open-saturated tableau (which indicates the existence of a model) and not a full model which should be even larger (see [50] for more details).

8 Conclusion

We defined in this article a new optimisation problem that we call the minimal S5 satisfiability problem (MinS5-SAT). It is the problem of finding the smallest S5-model w.r.t. the number of possible worlds. We demonstrated that this problem can be reduced to the problem of extracting a Maximal Satisfiability Set of clauses (MSS) and thus, can be solved with a MSS-extractor or one of the state-of-the-art PB or MaxSAT solvers. We also showed that, thanks to an inherent property of modal logic S5, this problem can also be solved using unsatisfiable cores in an incremental SAT procedure. The latter approach is the one that obtained the best performance in our experiments.

We applied these different techniques to various benchmarks: randomly generated formulas and also formulas expressing planning with uncertainties problems. Experimental results showed that the best technique for one set of benchmarks is not necessarily the best technique for the other, reminding us the importance of the choice of benchmarks in experimental evaluations. The technique used obtained huge gains in the size of the output models, when compared to the other approaches that do not try minimisation. In addition, the overhead imposed by the minimisation is acceptable. Therefore, we believe that finding minimal models for modal logic formulas is an interesting task. We can also mention that smaller models are more user-friendly, they permit to speedup the model checking phase and, in addition, some real-applications may prefer "smaller solutions" (smaller plans, for instance).

One possible future work is the application of these techniques to other NP-complete modal logics such as KD45 which is the belief counterpart of S5. Moreover, one could also try to solve the Minimal Satisfiability Problem without the use of a SAT solver, e.g. To compute the auto-bisimilar of a model retured by a Tableau proved such as Spartacus [48]. Also, one could try to solve the more general Minimal Modal Logic Satisfiability Problem (MinML-SAT), for which the standard satisfiability problem is typically PSPACE-hard. For instance, it would be interesting to try to filter the Minimal modal logic K satisfiability problem with the Minimal S5 satisfiability problem. It is known that satisfiability in S5 entails the satisfiability in K. If one finds a minimal S5-model of size n for a formula ϕ then the minimal K-model for ϕ has at most n possible worlds. This insight may help improving a naive search for the minimal K-model, because the only known bound b for modal logic K is an exponential function on the length the input formula.

Acknowledgments

We thank the anonymous reviewers for their insightful comments. Part of this work was supported by the French Ministry for Higher Education and Research, the Nord-Pas de Calais Regional Council through the "Contrat de Plan État Région (CPER) DATA" and an EC FEDER grant.

References

1. Fairtlough, M., Mendler, M.: An Intuitionistic Modal Logic with Applications to the Formal Verification of Hardware. In: Proc. of CSL'94. (1994) 354–368
2. Fitting, M.: Modality and Databases. In: Proc. of TABLEAUX'00. (2000) 19–39
3. Murphy VII, T., Crary, K., Harper, R.: Distributed Control Flow with Classical Modal Logic. In: Proc. of CSL'05. (2005) 51–69
4. Bienvenu, M., Fargier, H., Marquis, P.: Knowledge Compilation in the Modal Logic S5. In: Proc. of AAAI'10. (2010)
5. Niveau, A., Zanuttini, B.: Efficient Representations for the Modal Logic S5. In: Proc. of IJCAI'16. (2016) 1223–1229
6. Hustadt, U., Schmidt, R.A., Weidenbach, C.: MSPASS: Subsumption Testing with SPASS. In: Proc. of DL'99. Volume 22 of CEUR Workshop Proceedings., CEUR (1999)
7. Tacchella, A.: *SAT System Description. In: Proc. of DL'99. Volume 22., CEUR (1999)
8. Sebastiani, R., Vescovi, M.: Automated Reasoning in Modal and Description Logics via SAT Encoding: the Case Study of K(m)/ALC-Satisfiability. J.A.I.R **35** (2009) 343–389
9. Nalon, C., Hustadt, U., Dixon, C.: K_SP: A Resolution-Based Prover for Multimodal K. In: Proc. of IJCAR'16. (2016) 406–415
10. Caridroit, T., Lagniez, J.M., Le Berre, D., de Lima, T., Montmirail, V.: A SAT-Based Approach for Solving the Modal Logic S5-Satisfiability Problem. In: Proc. of AAAI'17. (2017)
11. Lagniez, J.M., Le Berre, D., de Lima, T., Montmirail, V.: A Recursive Shortcut for CEGAR: Application To The Modal Logic K Satisfiability Problem. In: Proc. of IJCAI'17. (2017)
12. Simon, L., Le Berre, D., Hirsch, E.A.: The SAT2002 Competition. Annals of Mathematics and A.I. **43**(1) (2005) 307–342
13. Marques-Silva, J., Janota, M.: On the Query Complexity of Selecting Few Minimal Sets. Electronic Colloquium on Computational Complexity (ECCC) **21** (2014) 31
14. Marques-Silva, J.P., Lynce, I.: On Improving MUS Extraction Algorithms. In: Proc. of SAT'11. (2011) 159–173
15. Belov, A., Marques-Silva, J.: Accelerating MUS Extraction With Recursive Model Rotation. In: Proc. of FMCAD'11. (2011) 37–40
16. Iser, M., Sinz, C., Taghdiri, M.: Minimizing Models for Tseitin-Encoded SAT Instances. In: Proc. of SAT'13. (2013) 224–232
17. Soh, T., Inoue, K.: Identifying Necessary Reactions in Metabolic Pathways by Minimal Model Generation. In: Proc. of ECAI'10. Volume 215., IOS Press (2010) 277–282
18. Koshimura, M., Nabeshima, H., Fujita, H., Hasegawa, R.: Minimal Model Generation With Respect To An Atom Set. In: Proc. of FTP'09. (2009) 49–59
19. Ladner, R.E.: The Computational Complexity of Provability in Systems of Modal Propositional Logic. SIAM J. of Computation **6**(3) (1977) 467–480
20. Cook, S.A.: Characterizations of Pushdown Machines in Terms of Time-Bounded Computers. J. of ACM **18**(1) (1971) 4–18
21. Li, C.M., Manyà, F.: MaxSAT, Hard and Soft Constraints. In: Handbook of Satisfiability. IOS Press (2009) 613–631
22. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning about knowledge. MIT Press (1995)
23. Eiter, T., Faber, W., Leone, N., Pfeifer, G., Polleres, A.: Planning under Incomplete Knowledge. In: Proc. of CL'00. (2000) 807–821
24. Kripke, S.A.: Semantical Analysis of Modal Logic I. Normal Propositional Calculi. Z.M.L.G.M **9**(56) (1963) 67–96
25. Eén, N., Sörensson, N.: An Extensible SAT-solver. In: Proc. of SAT'03. (2003) 502–518

26. Grégoire, É., Lagniez, J., Mazure, B.: An Experimentally Efficient Method for (MSS, CoMSS) Partitioning. In: Proc. of AAAI'14. (2014) 2666–2673
27. Belov, A., Lynce, I., Marques-Silva, J.: Towards Efficient MUS Extraction. AI Com. **25** (2012) 97–116
28. O'Sullivan, B., Papadopoulos, A., Faltings, B., Pu, P.: Representative Explanations for Over-Constrained Problems. In: Proc. of AAAI'07. (2007) 323–328
29. Audemard, G., Lagniez, J.M., Simon, L.: Improving Glucose for Incremental SAT Solving with Assumptions: Application to MUS Extraction. In: Proc. of SAT'13. (2013) 309–317
30. Roussel, O., Manquinho, V.M.: Pseudo-Boolean and Cardinality Constraints. In: Handbook of Satisfiability. IOS Press (2009) 695–733
31. Davies, J., Bacchus, F.: Exploiting the Power of MIP Solvers in MaxSAT. In: Proc. of SAT'13. (2013) 166–181
32. dos Reis Morgado, A.J., Ignatiev, A.S., Silva, J.M.: MSCG: Robust Core-Guided MaxSAT Solving. J. on Satisfiability Boolean Modeling and Computation **9** (2014) 129–134
33. Heras, F., Morgado, A., Marques-Silva, J.: Core-Guided Binary Search Algorithms for Maximum Satisfiability. In: Proc. of AAAI'11. (2011)
34. Sakai, M., Nabeshima, H.: Construction of an ROBDD for a PB-Constraint in Band Form and Related Techniques for PB-Solvers. IEICE Transactions **98-D**(6) (2015) 1121–1127
35. Le Berre, D., Parrain, A.: The SAT4J library, release 2.2. JSAT **7**(2-3) (2010) 59–6
36. Maher, S.J., Fischer, T., Gally, T., Gamrath, G., Gleixner, A., Gottwald, R.L., Hendel, G., Koch, T., Lübbecke, M.E., Miltenberger, M., Müller, B., Pfetsch, M.E., Puchert, C., Rehfeldt, D., Schenker, S., Schwarz, R., Serrano, F., Shinano, Y., Weninger, D., Witt, J.T., Witzig, J.: The SCIP Optimization Suite 4.0. Technical Report 17-12, ZIB, Takustr.7, 14195 Berlin (2017)
37. Mencía, C., Previti, A., Marques-Silva, J.: Literal-Based MCS Extraction. In: Proc. of IJCAI'15. (2015) 1973–1979
38. Josep Argelich and Chu Min Li and Felip Manyà and Jordi Planes: Max-SAT 2016: Eleventh Max-SAT Evaluation. http://www.maxsat.udl.cat/16/ (2016)
39. Saikko, P., Berg, J., Järvisalo, M.: LMHS: A SAT-IP Hybrid MaxSAT Solver. In: Proc. of SAT'16. (2016) 539–546
40. Massacci, F., Donini, F.M.: Design and Results of TANCS-2000 Non-classical (Modal) Systems Comparison. In: Proc. of TABLEAUX'00. (2000) 52–56
41. Kaminski, M., Tebbi, T.: InKreSAT: Modal Reasoning via Incremental Reduction to SAT. In: Proc. of CADE'13. (2013) 436–442
42. Balsiger, P., Heuerding, A., Schwendimann, S.: A Benchmark Method for the Propositional Modal Logics K, KT, S4. J. of Automated Reasoning **24**(3) (2000) 297–317
43. Patel-Schneider, P.F., Sebastiani, R.: A New General Method to Generate Random Modal Formulae for Testing Decision Procedures. J.A.I.R **18** (2003) 351–389
44. Clarke, E.M., Grumberg, O., Jha, S., Lu, Y., Veith, H.: Counterexample-Guided Abstraction Refinement for Symbolic Model Checking. J. of the ACM **50**(5) (2003) 752–794
45. Rintanen, J.: Planning and SAT. In: Handbook of Satisfiability. IOS Press (2009) 483–504
46. Petrick, R.P.A., Bacchus, F.: A Knowledge-Based Approach to Planning with Incomplete Information and Sensing. In: Proc. of AIPS'02. (2002) 212–222
47. Hoffmann, J., Brafman, R.I.: Conformant Planning via Heuristic Forward Search: A New Approach. A.I. **170**(6-7) (2006) 507–541
48. Götzmann, D., Kaminski, M., Smolka, G.: Spartacus: A Tableau Prover for Hybrid Logic. Electronic Notes in Theoretical Computer Science **262** (2010) 127–139
49. Weidenbach, C., Dimova, D., Fietzke, A., Kumar, R., Suda, M., Wischnewski, P.: SPASS Version 3.5. In: Proc. of CADE'09. (2009) 140–145
50. Lagniez, J.M., Le Berre, D., de Lima, T., Montmirail, V.: On Checking Kripke Models for Modal Logic K. In: Proc. of PAAR@IJCAR'16. (2016) 69–81