Artificial Intelligence 252 (2017) 22–50

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Minimal sets on propositional formulae. Problems and reductions

Joao Marques-Silva[a,*], Mikoláš Janota[b], Carlos Mencía[c]

[a] Department of Informatics, Faculty of Science, Univ. Lisbon, Portugal
[b] INESC-ID, IST, Univ. Lisbon, Portugal
[c] Department of Computer Science, University of Oviedo, Spain

A R T I C L E   I N F O

Article history:
Received 5 December 2015
Received in revised form 8 April 2017
Accepted 26 July 2017
Available online 2 August 2017

Keywords:
Boolean satisfiability
Minimal unsatisfiability
Maximal satisfiability
Minimal satisfiability
Maximal falsifiability
Irredundant subformulae
MUSes
MCSes
MESes
MFSes
Minimal and maximal models
Prime implicates and implicants
Formula entailment
Backbone literals
Variable independence
Autarkies
Independent support
Monotone predicates

A B S T R A C T

Boolean Satisfiability (SAT) is arguably the archetypical NP-complete decision problem. Progress in SAT solving algorithms has motivated an ever increasing number of practical applications in recent years. However, many practical uses of SAT involve solving function as opposed to decision problems. Concrete examples include computing minimal unsatisfiable subsets, minimal correction subsets, prime implicates and implicants, minimal models, backbone literals, and autarkies, among several others. In most cases, solving a function problem requires a number of adaptive or non-adaptive calls to a SAT solver. Given the computational complexity of SAT, it is therefore important to develop algorithms that either require the smallest possible number of calls to the SAT solver, or that involve simpler instances. This paper addresses a number of representative function problems defined on Boolean formulae, and shows that all these function problems can be reduced to a generic problem of computing a minimal set subject to a monotone predicate. This problem is referred to as the Minimal Set over Monotone Predicate (MSMP) problem. This work provides new ways for solving well-known function problems, including prime implicates, minimal correction subsets, backbone literals, independent variables and autarkies, among several others. Moreover, this work also motivates the development of more efficient algorithms for the MSMP problem. Finally the paper outlines a number of areas of future research related with extensions of the MSMP problem.

1. Introduction

The practical success of Propositional Satisfiability (SAT) solvers has motivated an ever increasing range of applications. Many applications are naturally formulated as decision procedures. In contrast, a growing number of applications involve solving function as opposed to decision problems. Representative examples include computing maximum satisfiability, minimum satisfiability, minimal unsatisfiable subsets of clauses, minimal correction subsets of clauses, minimal and maximal models, the backbone of a formula, the maximum autark assignment, prime implicants and implicates, among many others.

In recent years, the most widely used approach for solving a comprehensive range of function problems defined on Boolean formulae consists of using the SAT solver as an oracle, which is called a number of times polynomial in the

* Corresponding author.
E-mail address: jpms@acm.org (J. Marques-Silva).

http://dx.doi.org/10.1016/j.artint.2017.07.005
0004-3702/© 2017 Published by Elsevier B.V.

size of the problem representation. It is interesting to observe that this approach matches in some sense well-known query complexity characterizations of function problems [48,121]. Analysis of the solutions developed for different problems reveals similarities in the algorithms used, and often motivates the question whether a common set of algorithms could be considered instead. Among many other examples, this is apparent in algorithms for computing minimal unsatisfiable formulae and for computing the backbone of a propositional formula.

Moreover, as shown in [25,26,107], representative function problems can be viewed as computing a minimal set given some monotone predicate (MSMP), with the goal being to develop algorithms for computing a minimal set subject to a monotone predicate. This earlier work motivates the additional question of whether a more comprehensive set of function problems can be formulated as computing a minimal set subject to a monotone predicate. The consequences can be significant, both in terms of new algorithms for different problems, as well as possible new insights into some of these problems. This paper addresses this question, and shows that a large number of function problems defined on Boolean formulae can be cast as computing a minimal set subject to a monotone predicate.[1] As a result, the algorithms developed earlier [25,26,107] for some function problems can also be used for this much larger set of function problems. Similarly, algorithms specific to some function problems can be generalized and applied in the more general setting of computing a minimal set subject to a monotone predicate.

The main contributions of this paper can be summarized as follows. First, the paper formalizes the MSMP framework to a greater extent than earlier work [107]. Second, the paper identifies a large number of function problems that can be reduced to MSMP, and details one or more reductions in each case. Although the paper focuses on function problems defined on propositional formulae, the proposed ideas are readily applicable in other settings, including Constraint Satisfaction Problems (CSP) and Satisfiability Modulo Theories (SMT). Moreover, the presentation of the different reductions to MSMP outlines a methodology that could serve to easily identify additional function problems that can be reduced to the MSMP problem. Third, the paper develops a better understanding of the general properties of MSMP, including hitting set duality. Fourth and finally, the paper opens the avenue for developing generic MSMP solvers, capable of solving a wide range of function problems, and thereby contributing to eliminate the rediscovery of essentially the same algorithms in different settings.

It is important to note that recent work has already exploited relationships between different problems, enabling new insights for solving a growing number of function problems. Besides developing the MSMP framework and the progression algorithm, [107] shows that the QuickXplain [73] and FastDiag [47] algorithms correspond to the same algorithm, by changing only the predicate used and the argument to the predicate. A similar observation was made regarding the deletion-based algorithm [6,34] for finding minimal unsatisfiable subsets and the *grow* procedure [5,120] for finding minimal correction subsets. The relationship between minimal unsatisfiable subsets and minimal equivalent subsets was investigated in [10, 9]. Recent work showed that maximum autarkies could be computed with minimal correction subsets [104], and enabled developing new families of algorithms for computing maximum autarkies [89]. The problem of minimal independent support has been solved by reduction to computing minimal unsatisfiable subsets [65]. Monotonicity has been investigated in other settings including Satisfiability Modulo Theories (SMT) solving [7], but also for exploiting duality relationships [141]. Further relationships between problems that can be directly related with computing minimal correction subsets can now exploit the body of recent work on computing minimal correction subsets [103,3,50,114]. The same applies to computing the backbone of a propositional formula [69], not to mention the growing range of algorithms for computing minimal unsatisfiable subsets [118,152,11,4].

The paper is organized as follows. Section 2 introduces the notation and definitions used throughout the paper. Section 3 defines and overviews the function problems studied in later sections. Section 4 shows how all the function problems described in Section 3 can be reduced to solving an instance of the more general MSMP problem. This section also develops more fine-grained reductions, and proposing the aggregation of different function problems into related classes. Section 5 studies relationships between the different predicate forms used in previous sections, including hitting set duality. Section 6 discusses the practical impact of MSMP. Section 7 concludes the paper, by summarizing the main contributions and outlining a number of research directions.

2. Preliminaries

This section introduces the notation used in the remainder of the paper, covering propositional formulae, monotone predicates, complexity classes, and problem reductions.

2.1. Propositional formulae

Standard propositional logic definitions are used throughout the paper (e.g. [79,20]), some of which are reviewed in this section.

Sets are represented in calligraphic font, e.g. $\mathcal{R}, \mathcal{T}, \mathcal{I} \ldots$ Propositional formulae are also represented in calligraphic font, e.g. $\mathcal{F}, \mathcal{H}, \mathcal{S}, \mathcal{M} \ldots$ Propositional variables are represented with letters from the end of the alphabet, e.g. x, w, y, z, and

[1] A preliminary version of this work was made available as a CoRR report [105], building on an earlier conference paper [107]. This paper significantly extends and completes the earlier report and paper.

indices can be used, e.g. $x_1, w_1 \ldots$ An atom is a propositional variable. A literal is a variable x_i or its complement $\neg x_i$. A propositional formula[2] (or simply a formula) $\mathcal{F}$ is defined inductively over a set of propositional variables, with the standard logical connectives, $\neg$, $\wedge$, $\vee$, as follows:

1. An atom is a formula.
2. If $\mathcal{F}$ is a formula, then $(\neg\mathcal{F})$ is a formula.
3. If $\mathcal{F}$ and $\mathcal{G}$ are formulae, then $(\mathcal{F} \vee \mathcal{G})$ is a formula.
4. If $\mathcal{F}$ and $\mathcal{G}$ are formulae, then $(\mathcal{F} \wedge \mathcal{G})$ is a formula.

The inductive step could be extended to include other logical connectives, e.g. $\rightarrow$ and $\leftrightarrow$. (The use of parenthesis is not enforced, and standard binding rules apply, e.g. [79], with parenthesis being used only to clarify the presentation of formulae.) The inductive definition of a propositional formula allows for associating a parse tree with each formula, which can be used to evaluate truth assignments. The variables of a propositional formula $\mathcal{F}$ are represented by $\text{var}(\mathcal{F})$. For simplicity, the set of variables of a formula will be denoted by $X \triangleq \text{var}(\mathcal{F})$. A clause c is a non-tautologous disjunction of literals. A term t is a non-contradictory conjunction of literals. Commonly used representations of propositional formulae include conjunctive and disjunctive normal forms (resp. CNF and DNF). A CNF formula $\mathcal{F}$ is a conjunction of clauses. A DNF formula $\mathcal{F}$ is a disjunction of terms. CNF and DNF formulae can also be viewed as sets of sets of literals. Both representations will be used interchangeably throughout the paper. In the remainder of the paper, propositional formulae are referred to as formulae, and this can either represent an arbitrary propositional formula, a CNF formula, or a DNF formula. The necessary qualification will be used when necessary. The following sets are used throughout. $\mathbb{F}$ denotes the set of propositional formulae, $\mathbb{C} \subset \mathbb{F}$ denotes the set of CNF formulae, and $\mathbb{D} \subset \mathbb{F}$ denotes the set of DNF formulae.

When the set of variables of a formula $\mathcal{F}$ is relevant, the notation $\mathcal{F}[X]$ is used, meaning that F is defined in terms of variables from X. Replacements of variables will be used. The notation $\mathcal{F}[x_i/y_i]$ represents formula $\mathcal{F}$ with variable x_i replaced with y_i. This definition can be extended to more than one variable. For the general case, if $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$, then the notation $\mathcal{F}[X/Y]$ represents formula $\mathcal{F}$ with x_1 replaced by y_1, x_2 replaced by $y_2, \ldots$, and x_n replaced by y_n. Alternatively, one could write $\mathcal{F}[x_1/y_1, x_2/x_2, \ldots, x_n/y_n]$. Given $v_i \in \{0,1\}$, $\mathcal{F}[x_i/v_i]$ represents formula $\mathcal{F}$ with variable x_i replaced with v_i. Similarly, the notation $\mathcal{F}_{x_i=v_i}$ can be used.

The paper addresses mainly *plain* CNF formulae, i.e. formulae where any clause can be dropped (or relaxed) from the formula, being referred to as *soft* clauses [92]. Nevertheless, in some settings, CNF formulae can have *hard* clauses, i.e. clauses that cannot be dropped (or relaxed). In these cases, $\mathcal{F}$ can be viewed as a 2-tuple $\langle\mathcal{H}, \mathcal{B}\rangle$, where $\mathcal{H}$ denotes the hard clauses, and $\mathcal{B}$ denotes the *soft* (or relaxable, or breakable) clauses. (Observe that any satisfiability test involving $\mathcal{F}$ requires the hard clauses to be satisfied, whereas some of the clauses in $\mathcal{B}$ may discarded.) Moreover, weights can be associated with the soft clauses as follows. A weight function $\omega : \mathcal{H} \cup \mathcal{B} \rightarrow \mathbb{R} \cup \{\top\}$ associates a positive weight with each clause. For any soft clause $c \in \mathcal{B}$, $\omega(c) \neq \top$, whereas for any hard clause $c \in \mathcal{H}$, $\omega(c) = \top$, where $\top$ is such that it exceeds $\sum_{c\in\mathcal{B}} \omega(c)$, meaning that hard clauses are too costly to falsify. Throughout the paper, and unless otherwise stated, it is assumed that either $\mathcal{H} = \emptyset$ or $\mathcal{B} = \emptyset$, i.e. all clauses are soft or all clauses are hard, and so $\mathcal{F}$ corresponds to $\mathcal{B}$ or to $\mathcal{H}$, respectively. Moreover, the (implicit) weight function assigns cost 1 to each (soft) clause. Finally, the above definitions can be extended to handle groups of clauses, i.e. settings where formulae are partitioned into disjoint sets of clauses called *groups* (e.g. [99,57]).

Given a formula $\mathcal{F}$, a truth assignment ν is a map from the variables of $\mathcal{F}$ to $\{0,1\}$, $\nu : \text{var}(\mathcal{F}) \rightarrow \{0,1\}$. Given a truth assignment ν, the value taken by a formula, denoted $\mathcal{F}^\nu$ is defined inductively as follows:

1. If x is a variable, $x^\nu = \nu(x)$.
2. If $\mathcal{F} = (\neg\mathcal{G})$, then
$$\mathcal{F}^\nu = \begin{cases} 0 & \text{if } \mathcal{G}^\nu = 1 \\ 1 & \text{if } \mathcal{G}^\nu = 0 \end{cases}$$
3. If $\mathcal{F} = (\mathcal{E} \vee \mathcal{G})$, then
$$\mathcal{F}^\nu = \begin{cases} 1 & \text{if } \mathcal{E}^\nu = 1 \text{ or } \mathcal{G}^\nu = 1 \\ 0 & \text{otherwise} \end{cases}$$
4. If $\mathcal{F} = (\mathcal{E} \wedge \mathcal{G})$, then
$$\mathcal{F}^\nu = \begin{cases} 1 & \text{if } \mathcal{E}^\nu = 1 \text{ and } \mathcal{G}^\nu = 1 \\ 0 & \text{otherwise} \end{cases}$$

The inductive step could be extended to include additional propositional connectives, e.g. $\rightarrow$ and $\leftrightarrow$. A truth assignment ν such that $\mathcal{F}^\nu = 1$ is referred to as a *satisfying truth assignment*. A formula $\mathcal{F}$ is *satisfiable* if it has a satisfying truth assignment; otherwise it is *unsatisfiable*. As a result, the problem of propositional satisfiability is defined as follows:

[2] Throughout the paper, *propositional formula* and *Boolean formula* are used interchangeably.

Definition 1 *(Propositional Satisfiability (SAT)).* Given a formula $\mathcal{F}$, the decision problem SAT consists of deciding whether $\mathcal{F}$ is satisfiable.

The standard semantic entailment notation is used throughout. Let $\mathcal{A}, \mathcal{C} \in \mathbb{F}$. $\mathcal{A} \vDash \mathcal{C}$ denotes that for every truth assignment ν to the variables in $\text{var}(\mathcal{A}) \cup \text{var}(\mathcal{C})$, $(\mathcal{A}^\nu = 1) \Rightarrow (\mathcal{C}^\nu = 1)$. The equivalence notation $\mathcal{A} \equiv \mathcal{C}$ is used to denote that $\mathcal{A} \vDash \mathcal{C} \wedge \mathcal{C} \vDash \mathcal{A}$, indicating that $\mathcal{A}$ and $\mathcal{C}$ have the same satisfying truth assignments, when $\text{var}(\mathcal{A}) = \text{var}(\mathcal{C})$. In this case $\mathcal{A}$ and $\mathcal{C}$ are said to be *equisatisfiable*. If a formula $\mathcal{F} \in \mathbb{F}$ is satisfiable, we write $\mathcal{F} \nvDash \bot$. If a formula $\mathcal{F}$ is unsatisfiable, we write $\mathcal{F} \vDash \bot$. Moreover, if ν is a satisfying truth assignment of $\mathcal{F}$, the notation $\nu \vDash \mathcal{F}$ is also used. Finally, if $\mathcal{F}$ is a tautology, then the notation $\top \vDash \mathcal{F}$ is used, whereas $\top \nvDash \mathcal{F}$ denotes that $\mathcal{F}$ is not a tautology.

The following results are well-known, e.g. [79], and will be used throughout.

Proposition 1. *Let $\mathcal{F} \in \mathbb{C}$, with $\mathcal{F} \nvDash \bot$. Then, $\forall_{\mathcal{E} \in \mathbb{C}}, (\mathcal{E} \subseteq \mathcal{F}) \Rightarrow (\mathcal{E} \nvDash \bot)$.*

Proposition 2. *Let $\mathcal{U} \in \mathbb{C}$, with $\mathcal{U} \vDash \bot$. Then, $\forall_{\mathcal{T} \in \mathbb{C}}, (\mathcal{T} \supseteq \mathcal{U}) \Rightarrow (\mathcal{T} \vDash \bot)$.*

Proposition 3. *Let $\mathcal{F} \in \mathbb{D}$, with $\top \nvDash \mathcal{F}$. Then, $\forall_{\mathcal{E} \in \mathbb{D}}, (\mathcal{E} \subseteq \mathcal{F}) \Rightarrow (\top \nvDash \mathcal{E})$.*

Proposition 4. *Let $\mathcal{U} \in \mathbb{D}$, with $\top \vDash \mathcal{U}$. Then, $\forall_{\mathcal{T} \in \mathbb{D}}, (\mathcal{T} \supseteq \mathcal{U}) \Rightarrow (\top \vDash \mathcal{T})$.*

Propositions 1–4 are well-known, and follow from the compactness theorem. These will be used in the proofs provided in Section 4; however, since they are very well-known, they will not be explicitly mentioned for the sake of simplicity.

Given a formula $\mathcal{F}$, with set of variables $X = \text{var}(\mathcal{F})$, the following additional definitions apply. $\mathbb{L}(X) \triangleq \{x, \neg x \,|\, x \in X\}$ represents the set of literals given the variables in X. A truth assignment ν can be represented as a set of literals $\mathcal{V} \subseteq \mathbb{L}$ such that $|\mathcal{V}| = |X|$ and $\forall_{x \in X}, (x \notin \mathcal{V}) \vee (\neg x \notin \mathcal{V})$, interpreted as a conjunction of literals or a term, where each literal in $\mathcal{V}$ encodes the value assigned to a given variable $x \in X$, i.e. literal x if $\nu(x) = 1$ and literal $\neg x$ if $\nu(x) = 0$. In the remainder of the paper, an assignment can either be represented as a map or as a set of literals as defined above. This will be clear from the context.

Sets of literals are also used to represent partial truth assignments. The set of all partial truth assignments $\mathbb{A}$, given $X \triangleq \text{var}(\mathcal{F})$, is defined by $\mathbb{A}(X) \triangleq \{\mathcal{V} \subseteq \mathbb{L}(X) \mid (\forall_{x \in X}, (x \notin \mathcal{V}) \vee (\neg x \notin \mathcal{V}))\}$. (For simplicity, and when clear from the context, the dependency of $\mathbb{L}$ and $\mathbb{A}$ with X will be omitted.) Sets of literals can also be used to satisfy or falsify CNF or DNF formulae.

As noted above, sets of literals are in most cases interpreted as the conjunction of the literals, e.g. as a term. However, in some situations, it is convenient to interpret a set of literals as a disjunction of the literals, e.g. as a clause. Throughout the paper, the following convention is used. A set of literals qualified as a term, as a truth assignment, or as an implicant (defined below) is interpreted as a conjunction of literals. A set of literals qualified as a clause, or as an implicate (defined below) is interpreted as a disjunction of literals. Given $\mathcal{F} \in \mathbb{F}$, a term $t \in \mathbb{A}$ is an *implicant* of $\mathcal{F}$ iff $t \vDash \mathcal{F}$. (Alternatively, we could write $t \in \mathbb{A}$ is an *implicant* of $\mathcal{F}$ iff $\wedge_{l \in t} (l) \vDash \mathcal{F}$.) Similarly, a clause $c \in \mathbb{A}$ is an *implicate* of $\mathcal{F}$ iff $\mathcal{F} \vDash c$. (Similarly, we could write $c \in \mathbb{A}$ is an *implicate* of $\mathcal{F}$ iff $\mathcal{F} \vDash (\vee_{l \in c} l)$.)

In some settings it is necessary to reason with the variables assigned value 1. This is the case for example when reasoning with *minimal* and *maximal models*. Given a truth assignment, ν, with $\mathcal{V}$ the associated set of literals, the variables assigned value 1 are given by $\mathcal{M} = \mathcal{V} \cap X$. The function $\nu(\mathcal{M}, X)$ allows recovering the truth assignment associated with a set $\mathcal{M}$ of variables assigned value 1. If the assignment ν, associated with a set $\mathcal{M}$ of variables assigned value 1, is satisfying, then $\mathcal{M}$ is referred to as a *model*. Moreover, the notation $\nu(\mathcal{M}, X) \vDash \mathcal{F}$ is used to denote that, given a set $\mathcal{M}$ of variables assigned value 1, the associated truth assignment is satisfying. In contrast, $\nu(\mathcal{M}, X) \vDash \neg \mathcal{F}$ is used to denote that the truth assignment falsifies the formula.

Most of the algorithms described in this paper use sequences of calls to a SAT solver. A SAT solver accepts a (CNF-encoded) propositional formula $\mathcal{F}$ as a single argument and returns a 2-tuple (st, α), i.e. $(\text{st}, \nu) = \text{SAT}(\mathcal{F})$, where $\text{st} \in \{0, 1\}$ denotes whether the formula is unsatisfiable ($\text{st} = 0$, or simply unsat) or satisfiable ($\text{st} = 1$, or simply sat), and ν is a (satisfying) truth assignment in case the formula is satisfiable. ν will also be referred to as a *witness* of satisfiability. For simplicity, in this paper it is assumed the SAT solver does not return unsatisfiable subformulae when the outcome is unsat, although this feature is available in many modern SAT solvers, e.g. [43,19]. Throughout the paper, it will be implicitly assumed that a SAT solver call yields not only the 0/1 outcome, but the actual 2-tuple (st, ν).

Modern SAT solvers typically accept CNF formulae [110]. Procedures for CNF-encoding (or clausifying) arbitrary propositional formulae are well-known (e.g. [146,122]). These procedures run in linear time on the size of the formula, and generate a new formula that is linear on the size of the original formula, and which uses at most a linear number of additional new variables. Throughout the paper the propositional formulae passed to SAT solvers are often not in CNF. For simplicity, it is left implicit that a clausification procedure would be invoked if necessary. Moreover, additional non-clausal constraints will be used. These include pseudo-Boolean constraints and cardinality constraints, e.g. [132,123,44]. Examples of clausification approaches are described in [123,44].

2.2. Monotone predicates & MSMP

Given a predicate $P : 2^{\mathcal{R}} \to \{0, 1\}$, defined on a reference set $\mathcal{R}$, a minimal set over P is defined as follows.

Algorithm 1: Deletion-based computation of a minimal set.

```
Function DELETION ($P$,$\mathcal{R}$)
  Input: $P$: Monotone predicate, $\mathcal{R}$: Reference set
  Output: $\mathcal{M}$: Minimal set
1 $\mathcal{M} \leftarrow \mathcal{R}$                          // Minimal set over-approximation
2 foreach $u \in \mathcal{M}$ do                               // Inv: $P(\mathcal{M})$
3     if $P(\mathcal{M} \setminus \{u\})$ then                 // $u$ is not in minimal set
4         $\mathcal{M} \leftarrow \mathcal{M} \setminus \{u\}$  // Drop element
5 return $\mathcal{M}$                                         // Final $\mathcal{M}$ is a minimal set
```

Definition 2. Let P be a predicate, and let $\mathcal{M} \subseteq \mathcal{R}$ such that $P(\mathcal{M})$ holds. $\mathcal{M}$ is *minimal* over P if and only if $\forall_{\mathcal{M}' \subsetneq \mathcal{M}}, \neg P(\mathcal{M}')$.

This paper focuses on a special kind of predicates, referred to as *monotone* (e.g. [25]).

Definition 3. A predicate P, defined on a set $\mathcal{R}$, is said to be *monotone* if whenever $P(\mathcal{R}_0)$ holds, with $\mathcal{R}_0 \subseteq \mathcal{R}$, then $P(\mathcal{R}_1)$ also holds, with $\mathcal{R}_0 \subseteq \mathcal{R}_1 \subseteq \mathcal{R}$.

Observe that $P(\mathcal{R})$ can be assumed, but this is not required. Also, $P(\mathcal{R})$ can be tested with a single predicate test. Moreover, observe that, if there exists a set $\mathcal{R}_0 \subseteq \mathcal{R}$ such that $P(\mathcal{R}_0)$ holds, and P is monotone, then $P(\mathcal{R})$ also holds. Otherwise, no minimal set exists.

Example 1. It is simple to conclude that, given a finite set $\mathcal{W}$, predicate $P(\mathcal{W}) \triangleq [\,|\mathcal{W}| \geq K\,]$, for some $K \geq 0$, is monotone. In contrast, predicate $P(\mathcal{W}) \triangleq [\,|\mathcal{W}| \bmod 2 = 1\,]$ is not monotone.

Minimality over a monotone predicate can be tested without the need of considering all the proper subsets of a given set. This is captured by the following lemma, which will be used throughout.

Lemma 1. *Let P be a monotone predicate, $\mathcal{R}$ a set such that $P(\mathcal{R})$ holds, and $\mathcal{M} \subseteq \mathcal{R}$. $\mathcal{M}$ is a minimal set over P if $P(\mathcal{M})$ holds and $\forall_{u \in \mathcal{M}}, \neg P(\mathcal{M} \setminus \{u\})$.*

Proof. Suppose $P(\mathcal{M})$ holds, $\forall_{u \in \mathcal{M}}, \neg P(\mathcal{M} \setminus \{u\})$ and $\mathcal{M}$ is not minimal. Then there must be a set $\mathcal{M}' \subsetneq \mathcal{M}$ s.t. $P(\mathcal{M}')$ holds. Since P is monotone, for all $\mathcal{M}''$ s.t. $\mathcal{M}' \subseteq \mathcal{M}'' \subseteq \mathcal{M}$, $P(\mathcal{M}'')$ holds. Therefore, there exists $u \in \mathcal{M}$ s.t. $P(\mathcal{M} \setminus \{u\})$ holds. A contradiction. □

Definition 4 *(MSMP Problem).* Given a monotone predicate P, the *Minimal Set over a Monotone Predicate* (MSMP) problem consists in finding a minimal subset $\mathcal{M}$ of $\mathcal{R}$ such that $P(\mathcal{M})$ holds.

Monotone predicates were used in [25,26] to describe algorithms for computing a prime implicate of a propositional formula given a clause. The MSMP problem was introduced in [107]. The same work [107] also showed that a number of additional function problems defined on propositional formulae could be reduced to the MSMP problem, including the function problems of computing minimal unsatisfiable subsets, minimal correction subsets and minimal models.

Algorithm 1 for the MSMP problem illustrates the practical relevance of this framework. It shows what is referred to as the *deletion-based* approach. Given a monotone predicate P and a reference set $\mathcal{R}$, it iteratively refines an upper-approximation $\mathcal{M} \subseteq \mathcal{R}$ of a minimal set over P. Initially, $\mathcal{M}$ is set to $\mathcal{R}$. Then, for each $u \in \mathcal{M}$, the algorithm tests whether u can be dropped from $\mathcal{M}$ so that the predicate still holds. If this is the case, the element u is dropped from $\mathcal{M}$. Otherwise it is kept. Eventually, $P(\mathcal{M})$ holds, and for all $u \in \mathcal{M}$, $P(\mathcal{M} \setminus \{u\})$ does not hold, i.e. $\mathcal{M}$ is a minimal set over P.

Deletion-based approaches have been extensively used to compute minimal unsatisfiable subsets (MUSes) [6], and irreducible infeasible constraint sets [34]. The deletion-based algorithm has a query complexity of $\Theta(n)$ predicate tests, with $n = |\mathcal{R}|$. This same algorithm can also be shown to correspond to the *grow* procedure [5] or linear search procedure for computing minimal correction subsets (MCSes) [120,103].

As shown in earlier work [107], different MUS extraction algorithms can be generalized to MSMP, and so can be used for a wide range of function problems. Besides deletion-based, shown above, other algorithms include *insertion* [140], QuickXplain [73], *dichotomic* [55], and *progression* [107,90].[3] Moreover, other MUS extraction algorithms [109] can be easily adapted.

[3] It should be noted that the insights used in the *progression* MSMP algorithm can be traced to work on finding extremal sets [126], but also to earlier work on saddleback search [42] and unbounded search [15]. Similarly, the QuickXplain algorithm can be related with work on Delta Debugging [154].

2.3. Function problems

For many computational problems, the main goal is not to solve a decision problem, but to solve instead a *function problem* [121, Section 10.3].[4] The goal of a function problem is to compute some solution, e.g. a satisfying truth assignment in the case of SAT, the size of a prime implicate, the actual prime implicate, the largest number of clauses that any truth assignment can satisfy in a CNF formula, etc. This paper addresses function problems defined on propositional formulae, which are defined in Section 3. The main focus of the paper are function problems that can be related with computing a minimal set over a monotone predicate, i.e. function problems that can be reduced to the MSMP problem.

2.4. Problem reductions

The notation $A \leq_p B$ is used to denote that any instance of a function problem A can be solved by solving instead a polynomially-related instance of another function problem B. In this paper, B denotes the MSMP problem, and any reduction $A \leq_p B$ indicates that the solution to the function problem A for a concrete instance a can be obtained by computing a minimal set (subject to a monotone predicate) for some resulting concrete instance b of the MSMP problem. Moreover, all reductions described in the paper are provided with enough detail to make it straightforward to conclude that the problem instances are polynomially related.

2.5. Related work

The work in this paper is motivated by the use of monotone predicates for computing a prime implicate of a propositional formula given an implicate (e.g. a clause) [25,26]. This work was recently extended to show that monotone predicates can be applied to other problems, namely MUSes, MCSes, minimal models and (also) prime implicates given a clause [107]. The same work, [107], also proposed the *progression* algorithm for the MSMP problem. The work in [107] also enabled a characterization of the query complexity of different function problems [106,70], with new results for the case of computing the backbone of a formula or computing a minimal unsatisfiable subset, when the number of these is constant.

A tightly related concept in computational complexity is *hereditarity* [32,33], which was proposed in the 90s to develop lower and upper bounds on the query complexity of computing maximal solutions. A concrete example is the computation of minimal unsatisfiability and maximal satisfiability [32,33]. In contrast, other function problems, e.g. minimal and maximal models, involve different definitions from those standard in AI (e.g. [13]). Moreover, one concern of function problem solving with SAT solvers is to develop a rigorous characterization of a SAT solver as an oracle. As argued elsewhere [106], if the time spent on the SAT solver is ignored, then a SAT solver can viewed as a witness oracle [29].

The paper addresses function problems defined on Boolean formulae, which intersect a wide range of areas of research. References to related work, both on the actual function problems, and associated areas of research are included throughout the paper.

Finally, as indicated in Section 1, monotone predicates have already influenced a number of areas of research. Besides algorithms for function problem solving, other investigated topics include monotonicity in general settings [7] and duality also in general settings [141].

3. Target function problems

This section provides a brief overview of the function problems studied in the rest of the paper. All function problems studied in this section are defined on Boolean formulae. Nevertheless, the work can be extended to function problems defined on more expressive domains, e.g. ILP, SMT, CSP, etc.

3.1. Problem definitions

The function problems described below can be organized as follows: (i) minimal unsatisfiability (and maximal satisfiability); (ii) irredundant subformulae; (iii) maximal falsifiability (and minimal satisfiability); (iv) minimal and maximal models; (v) prime implicates and implicants; (vi) backbone literals; (vii) formula entailment; (viii) variable independence; (ix) maximum autarkies; and (x) minimal independent support. Afterwards, Section 4 shows that all these function problems can be reduced to the MSMP problem. In what follows, and unless otherwise stated, $\mathcal{F}$ denotes an arbitrary Boolean formula. In some specific cases, $\mathcal{F}$ is restricted to be in CNF (or in DNF), and this will be noted.

3.1.1. Minimal unsatisfiability & maximal satisfiability

Minimal unsatisfiability and maximal satisfiability have been extensively studied in a number of contexts (e.g. [78,92,11,117] and references therein). In this paper, the following definitions are used.

Definition 5 *(MUS; FMUS).* Let $\mathcal{F} \in \mathbb{C}$, with $\mathcal{F} \vDash \perp$. $\mathcal{M} \subseteq \mathcal{F}$ is a *Minimal Unsatisfiable Subset* (MUS) iff $\mathcal{M} \vDash \perp$ and $\forall_{\mathcal{M}' \subsetneq \mathcal{M}}, \mathcal{M}' \nvDash \perp$. FMUS is the function problem of computing an MUS of $\mathcal{F}$.

[4] Function problems are also referred to as *search problems* [48, Section 5.1].

Definition 6 *(MCS; FMCS).* Let $\mathcal{F} \in \mathbb{C}$, with $\mathcal{F} \vDash \bot$. $\mathcal{C} \subseteq \mathcal{F}$ is a *Minimal Correction Subset* (MCS) iff $\mathcal{F} \setminus \mathcal{C} \nvDash \bot$ and $\forall_{\mathcal{C}' \subsetneq \mathcal{C}}, \mathcal{F} \setminus \mathcal{C}' \vDash \bot$. FMCS is the function problem of computing an MCS of $\mathcal{F}$.

Definition 7 *(MSS; FMSS).* Let $\mathcal{F} \in \mathbb{C}$, with $\mathcal{F} \vDash \bot$. $\mathcal{S} \subseteq \mathcal{F}$ is a *Maximal Satisfiable Subset* (MSS) iff $\mathcal{S} \nvDash \bot$ and $\forall_{\mathcal{F} \supseteq \mathcal{S}' \supsetneq \mathcal{S}}, \mathcal{S}' \vDash \bot$. FMSS is the function problem of computing an MSS of $\mathcal{F}$.

In short, an MUS is an unsatisfiable subset of the clauses of $\mathcal{F}$ that becomes satisfiable whenever any of its clauses are dropped from it. An MCS is a subset of the clauses of $\mathcal{F}$ whose removal renders $\mathcal{F}$ satisfiable, but such that adding any of these clauses back into $\mathcal{F}$ makes it unsatisfiable. An MSS is the complement of an MCS. This relationship is well-known, e.g. [21,99]:

Remark 1. $\mathcal{C}$ is an MCS of $\mathcal{F}$ iff $\mathcal{S} = \mathcal{F} \setminus \mathcal{C}$ is an MSS of $\mathcal{F}$.

Example 2. Let $\mathcal{F} = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (x_1), (\neg x_1)\}$. $\mathcal{F}$ has two MUSes: $\mathcal{M}_1 = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (\neg x_1)\}$ and $\mathcal{M}_2 = \{(x_1), (\neg x_1)\}$. The MCSes of $\mathcal{F}$ are $\mathcal{C}_1 = \{(\neg x_1)\}$, $\mathcal{C}_2 = \{(x_1 \vee x_2), (x_1)\}$ and $\mathcal{C}_3 = \{(x_1 \vee \neg x_2), (x_1)\}$. The MSSes of $\mathcal{F}$ are $\mathcal{S}_1 = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (x_1)\}$, $\mathcal{S}_2 = \{(x_1 \vee \neg x_2), (\neg x_1)\}$ and $\mathcal{S}_3 = \{(x_1 \vee x_2), (\neg x_1)\}$.

The MaxSAT problem is defined assuming unweighted clauses.

Definition 8 *(LMSS; FLMSS/MaxSAT).* Let $\mathcal{F} \in \mathbb{C}$, with $\mathcal{F} \vDash \bot$. An MSS $\mathcal{S}$ of $\mathcal{F}$ is a largest MSS (LMSS) iff for any MSS $\mathcal{S}'$ of $\mathcal{F}$, $|\mathcal{S}'| \leq |\mathcal{S}|$. FLMSS (or MaxSAT) is the function problem of computing an LMSS of $\mathcal{F}$.

Observation 1. Observe that MaxSAT is defined as the function problem of computing one largest MSS. In some contexts, other definitions are used, namely as the problem of computing the largest number of simultaneously satisfied clauses [83]. This distinction is quite significant for the unweighted case of MaxSAT. The definition used in this paper aims to model the function problem actually solved by modern MaxSAT solvers. Since practical MaxSAT solvers always compute a witness of the reported solution, this avoids the issue with reproducing a witness in the case an NP oracle was considered.

Definition 9 *(SMCS; FSMCS).* Let $\mathcal{F} \in \mathbb{C}$, with $\mathcal{F} \vDash \bot$. An MCS $\mathcal{C}$ of $\mathcal{F}$ is a smallest MCS (SMCS) iff for any MCS $\mathcal{C}'$ of $\mathcal{F}$, $|\mathcal{C}'| \geq |\mathcal{C}|$. FSMCS is the function problem of computing an SMCS of $\mathcal{F}$.

Remark 2. Clearly, $\mathcal{S} \subseteq \mathcal{F}$ is an LMSS of $\mathcal{F}$ iff $\mathcal{C} = \mathcal{F} \setminus \mathcal{S}$ is an SMCS of $\mathcal{F}$.

Moreover, it is straightforward to extend the above definitions to cases where there are hard clauses (i.e. clauses that cannot be dropped from the formula) and where soft clauses have weights [92]. There exists a large body of theoretical and practical work on computing MUSes, MCSes and on solving MaxSAT, including a well-known minimal hitting set duality relationship between MUSes and MCSes [129,21,99]. Recent references describing new algorithms and surveys for these problems include [52,92,102,11,1,103,117,50,3,114]. There is a growing number of practical applications of both minimal unsatisfiability and maximal satisfiability (e.g. see [52,102,11,117] and references therein). Recent examples of applications include design debugging [133], fault localization [71,72], model checking [81,54] or interactive reconfiguration [68]. Extraction of MUSes for non-clausal formulae and for SMT formulae is addressed in [78,12,155,81,68,54].

Finally, we should note that similar definitions can be developed for DNF formulae, by considering *minimal validity* instead of *minimal unsatisfiability*. This would also allow developing several tightly related function problems, as above.

MSSes with assumptive contexts A problem tightly related with finding MSSes is that of computing MSSes given assumptive contexts, which may be mutually contradictory [16]. A number of practical applications of finding MSSes given assumptive contexts is covered in past work [16].

Definition 10 *(MSSes with Assumptive Contexts; FACMCS).* Let $\mathcal{F} \in \mathbb{C}$ and $\Gamma \subsetneq \mathbb{F}$, with Γ finite. $\mathcal{M}$ is a maximal satisfiable subset of $\mathcal{F}$ under a set of assumptive contexts Γ, i.e. $\mathcal{M}$ is an $\text{AC} - \text{MSS}(\mathcal{F}, \Gamma)$ iff,

- $\forall_{\gamma \in \Gamma} \mathcal{M} \cup \{\gamma\} \nvDash \bot$.
- $\forall_{c \in \mathcal{F} \setminus \mathcal{M}} \exists_{\gamma \in \Gamma} \mathcal{M} \cup \{c\} \cup \{\gamma\} \vDash \bot$.

The complement of an $\text{AC} - \text{MSS}$ $\mathcal{M}$ is denoted an $\text{AC} - \text{MCS}$.

Observe that earlier work [16] also requires $\mathcal{M} \subseteq \mathcal{F} \nvDash \bot$, but this condition is redundant given the other ones. Similarly, the definition above can be extended to the concept of multiple contraction [51], if cardinality minimality is replaced by subset minimality. Multiple contraction of set of clauses $\mathcal{F}$ by a set of formulae Γ is a maximum cardinality subset of $\mathcal{F}$ from which no formula of Γ can be deduced. This implies that MSMP can also be applied in the context of multiple contraction, again provided that cardinality minimality is replaced by subset minimality. In addition, the same comment can be made for anti-subsumptive knowledge enforcement [49].

3.1.2. Irredundant subformulae

The definitions for minimal unsatisfiability (see previous section) can be reformulated for the case where the goal is to remove redundancy from a CNF formula, as follows.

Definition 11 *(MES; FMES).* Let $\mathcal{F} \in \mathbb{C}$. $\mathcal{E} \subseteq \mathcal{F}$ is a *Minimal Equivalent Subset* (MES) iff $\mathcal{E} \equiv \mathcal{F}$ and $\forall_{\mathcal{E}' \subsetneq \mathcal{E}}, \mathcal{E}' \not\equiv \mathcal{F}$. FMES is the function problem of computing an MES of $\mathcal{F}$.

Definition 12 *(MDS; FMDS).* Let $\mathcal{F} \in \mathbb{C}$. $\mathcal{D} \subseteq \mathcal{F}$ is a *Minimal Distinguishing Subset* (MDS) iff $\mathcal{F} \setminus \mathcal{D} \not\equiv \mathcal{F}$ and $\forall_{\mathcal{D}' \subsetneq \mathcal{D}}, \mathcal{F} \setminus \mathcal{D}' \equiv \mathcal{F}$. FMDS is the function problem of computing an MDS of $\mathcal{F}$.

Definition 13 *(MNS; FMNS).* Let $\mathcal{F} \in \mathbb{C}$. $\mathcal{N} \subseteq \mathcal{F}$ is a *Maximal Non-equivalent Subset* (MNS) iff $\mathcal{N} \not\equiv \mathcal{F}$ and $\forall_{\mathcal{F} \supseteq \mathcal{N}' \supsetneq \mathcal{N}}, \mathcal{N}' \equiv \mathcal{F}$. FMNS is the function problem of computing an MNS of $\mathcal{F}$.

An MES can be seen as a generalization of an MUS. It is a subset of clauses of $\mathcal{F}$ that is equivalent to $\mathcal{F}$ but dropping any of its clauses makes it non-equivalent to $\mathcal{F}$. Similarly, an MDS (MNS) can be seen as a generalization of an MCS (MSS). An MDS is a subset of the clauses of $\mathcal{F}$ whose removal makes the formula non-equivalent to what initially was, but adding any of these clauses back into $\mathcal{F}$ makes it equivalent. As with MCSes/MSSes, an MNS is the complement of an MDS, and vice-versa.

Remark 3. $\mathcal{D}$ is an MDS of $\mathcal{F}$ iff $\mathcal{N} = \mathcal{F} \setminus \mathcal{D}$ is an MNS of $\mathcal{F}$.

Example 3. Let $\mathcal{F} = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (x_1)\}$. The MESes of $\mathcal{F}$ are $\mathcal{E}_1 = \{(x_1 \vee x_2), (x_1 \vee \neg x_2)\}$ and $\mathcal{E}_2 = \{(x_1)\}$. The MDSes of $\mathcal{F}$ are $\mathcal{D}_1 = \{(x_1 \vee x_2), (x_1)\}$ and $\mathcal{D}_2 = \{(x_1 \vee \neg x_2), (x_1)\}$. The MNSes of $\mathcal{F}$ are $\mathcal{N}_1 = \{(x_1 \vee \neg x_2)\}$ and $\mathcal{N}_2 = \{(x_1 \vee x_2)\}$.

This paper also considers the optimization version of FMDS (FMNS), that is, computing a cardinality minimal MDS (cardinality maximum MDS). As before, these problems are defined considering unweighted clauses.

Definition 14 *(LMNS; FLMNS).* Let $\mathcal{F} \in \mathbb{C}$. An MNS $\mathcal{N}$ of $\mathcal{F}$ is a largest MNS (LMNS) iff for any MNS $\mathcal{N}'$ of $\mathcal{F}$, $|\mathcal{N}'| \leq |\mathcal{N}|$. FLMNS is the function problem of computing an LMNS of $\mathcal{F}$.

Definition 15 *(SMDS; FSMDS).* Let $\mathcal{F} \in \mathbb{C}$. An MDS $\mathcal{D}$ of $\mathcal{F}$ is a smallest MDS (SMDS) iff for any MDS $\mathcal{D}'$ of $\mathcal{F}$, $|\mathcal{D}'| \geq |\mathcal{D}|$. FSMDS is the function problem of computing an SMDS of $\mathcal{F}$.

Remark 4. Clearly, $\mathcal{N} \subseteq \mathcal{F}$ is an LMNS of $\mathcal{F}$ iff $\mathcal{D} = \mathcal{F} \setminus \mathcal{N}$ is an SMDS of $\mathcal{F}$.

Moreover, it is straightforward to extend the above definitions to cases where there are hard clauses (i.e. clauses that cannot be dropped from the formula) and where soft clauses have weights.

Complexity characterizations of computing irredundant subformulae were studied in [94–96,87]. Recent practical algorithms include [24,35,9,10].

Finally, we should note that definitions of irredundancy can be developed for DNF formulae in terms of a subset-minimal set of terms equivalent to the original formula. As before, this would allow defining additional function problems, as above.

3.1.3. Minimal satisfiability & maximal falsifiability

In some settings, the goal is to minimize the number of satisfied clauses. This is generally referred to as the *Minimum Satisfiability* (MinSAT) problem [80]. By analogy with the MaxSAT case, one can also consider extremal sets [62].

Definition 16 *(All-Falsifiable).* Let $\mathcal{F} \in \mathbb{C}$. $\mathcal{U} \subseteq \mathcal{F}$ is *All-Falsifiable* if there exists a truth assignment ν, to $\text{var}(\mathcal{U})$, such that ν falsifies all clauses in $\mathcal{U}$.

Definition 17 *(MFS; FMFS).* Given $\mathcal{F} \in \mathbb{C}$, a *Maximal Falsifiable Subset* (MFS) of $\mathcal{F}$ is a set $\mathcal{M} \subseteq \mathcal{F}$ such that $\mathcal{M}$ is all-falsifiable and, $\forall_{\mathcal{F} \supseteq \mathcal{N} \supsetneq \mathcal{M}}, \mathcal{N}$ is not all-falsifiable. FMFS is the function problem of computing an MFS of $\mathcal{F}$.

Definition 18 *(MCFS; FMCFS).* Given $\mathcal{F} \in \mathbb{C}$, a *Minimal Correction (for Falsifiability) Subset* (MCFS) is a set $\mathcal{C} \subseteq \mathcal{F}$ such that $\mathcal{F} \setminus \mathcal{C}$ is all-falsifiable and $\forall_{\mathcal{C}' \subsetneq \mathcal{C}}, \mathcal{F} \setminus \mathcal{C}'$ is not all-falsifiable. FMCFS is the function problem of computing an MCFS of $\mathcal{F}$.

Remark 5. $\mathcal{M}$ is an MFS of $\mathcal{F}$ iff $\mathcal{N} = \mathcal{F} \setminus \mathcal{D}$ is an MCFS of $\mathcal{F}$.

Example 4. Let $\mathcal{F} = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (x_1 \vee x_3)\}$. $\mathcal{F}$ has two MFSes: $\mathcal{M}_1 = \{(x_1 \vee x_2), (x_1 \vee x_3)\}$ and $\mathcal{M}_2 = \{(x_1 \vee \neg x_2), (x_1 \vee x_3)\}$. The MCFSes of $\mathcal{F}$ are $\mathcal{C}_1 = \{(x_1 \vee \neg x_2)\}$ and $\mathcal{C}_2 = \{(x_1 \vee x_2)\}$.

As before, optimization problems for maximum falsifiability and minimum satisfiability are considered.

Definition 19 *(LMFS; FLMFS/MaxFalse).* Let $\mathcal{F} \in \mathbb{C}$. An MFS $\mathcal{M}$ of $\mathcal{F}$ is a largest MFS (LMFS) iff for any MFS $\mathcal{M}'$ of $\mathcal{F}$, $|\mathcal{M}'| \leq |\mathcal{M}|$. FLMFS (or *Maximum Falsifiability*, MaxFalse) is the function problem of computing an LMFS of $\mathcal{F}$.

Definition 20 *(SMCFS; FSMCFS/MinSAT).* Let $\mathcal{F} \in \mathbb{C}$. An MCFS $\mathcal{C}$ of $\mathcal{F}$ is a smallest MCFS (SMCFS) iff for any MCFS $\mathcal{C}'$ of $\mathcal{F}$, $|\mathcal{C}'| \geq |\mathcal{C}|$. FSMCFS (or *Minimum Satisfiability*, MinSAT) is the function problem of computing an SMCFS of $\mathcal{F}$.

Remark 6. Clearly, $\mathcal{M} \subseteq \mathcal{F}$ is an LMFS of $\mathcal{F}$ iff $\mathcal{C} = \mathcal{F} \setminus \mathcal{M}$ is an SMCFS of $\mathcal{F}$.

Remark 7. Given Remark 5, one can conclude that the MinSAT problem consists of computing the smallest MCFS, and this represents the complement of the MaxFalse solution. Thus, for $\mathcal{F} \in \mathbb{C}$, where n_t and n_f denote respectively the MinSAT and the MaxFalse solutions, then $|\mathcal{F}| = n_t + n_f$.

As with minimal unsatisfiability and irredundancy, maximal falsifiability can be generalized to the case when some clauses are hard and when soft clauses have weights. Similarly, one could consider the DNF formulae, and function problems related with *all-true* terms.

The MinSAT problem has been studied in [80,101,2,93,61]. The problem of maximal falsifiability is studied in [62].

Moreover, it should be noted that, although this paper addresses MUSes/MCSes/MSSes/MESes/MDSes/MNSes/MFSes/-MCFSes defined solely on sets of clauses, other variants could be considered, namely groups of clauses or variables. The formalizations of these variants as instances of the MSMP problem mimic the formalizations developed for the case of sets of clauses. (See [99,40,118,56,8,57] for related work on groups of clauses and variables.)

3.1.4. Minimal & maximal models

Minimal and maximal models are additional examples of function problems associated with propositional formulae.

Definition 21 *(Minimal Model; FMnM).* Given $\mathcal{F} \in \mathbb{F}$, a model $\mathcal{M} \subseteq X$ of $\mathcal{F}$ is minimal iff $\nu(\mathcal{M}, X) \vDash \mathcal{F}$ and $\forall_{\mathcal{M}' \subsetneq \mathcal{M}}$, $\nu(\mathcal{M}', X) \nvDash \mathcal{F}$. FMnM is the function problem of computing a minimal model of $\mathcal{F}$.

Definition 22 *(Maximal Model; FMxM).* Given $\mathcal{F} \in \mathbb{F}$, a model $\mathcal{M} \subseteq X$ of $\mathcal{F}$ is maximal iff $\nu(\mathcal{M}, X) \vDash \mathcal{F}$ and $\forall_{X \supseteq \mathcal{M}' \supsetneq \mathcal{M}}$, $\nu(\mathcal{M}', X) \nvDash \mathcal{F}$. FMxM is the function problem of computing a maximal model of $\mathcal{F}$.

Example 5. Let $\mathcal{F} = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (x_2 \vee x_3)\}$. The models $\mathcal{M}_1 = \{x_1, x_2\}$ and $\mathcal{M}_2 = \{x_1, x_3\}$ are minimal models. The model $\mathcal{M}_3 = \{x_1, x_2, x_3\}$ is a maximal model of $\mathcal{F}$.

Observation 2. Both minimal and maximal models can be computed subject to a set Z of variables other than $X \triangleq \text{var}(\mathcal{F})$ [13], i.e. the so-called Z-minimal and Z-maximal models. The above definitions can easily be modified for this more general definition.

Minimal models find a wide range of applications, including non-monotonic reasoning and bioinformatics (e.g. [45,143]). Algorithms for computing minimal and maximal models have been studied in the past (e.g. [13,119,14,75,82]). In some contexts, minimal and maximal models have been referred to as MIN-ONE$_{\subseteq}$ and MAX-ONE$_{\subseteq}$ solutions (e.g. [131]).

As before, we can define smallest minimal models.

Definition 23 *(SMnM; FSMnM).* Let $\mathcal{F} \in \mathbb{F}$, with $\mathcal{F} \nvDash \bot$. A minimal model $\mathcal{M}$ of $\mathcal{F}$ is a smallest minimal model (SMnM) iff for any minimal model $\mathcal{M}'$ of $\mathcal{F}$, $|\mathcal{M}'| \geq |\mathcal{M}|$. FSMnM is the function problem of computing an SMnM of $\mathcal{F}$.

The definition of largest maximal model mimics the one above.

3.1.5. Implicants & implicates

Two relevant function problems associated with propositional formulae consist of computing prime implicants, starting from an implicant represented as a term, and prime implicates, starting from an implicate represented as a clause.[5]

Definition 24 *(Prime Implicant (given term); FPIt).* Given $\mathcal{F} \in \mathbb{F}$ and an implicant $t \in \mathbb{A}$ of $\mathcal{F}$, i.e. $t \vDash \mathcal{F}$, term $u \subseteq t$ is a prime implicant of $\mathcal{F}$ iff $u \vDash \mathcal{F}$ and $\forall_{v \subsetneq u}$, $v \nvDash \mathcal{F}$. FPIt is the function problem of computing a prime implicant of $\mathcal{F}$ given an implicant t of $\mathcal{F}$.

[5] The definitions of prime implicant and implicate used in this paper follow a wide range of sources, e.g. [127,33,147]. Alternative definitions have been considered in the past [111].

Definition 25 *(Prime Implicate (given clause); FPIc).* Given $\mathcal{F} \in \mathbb{F}$ and an implicate $c \in \mathbb{A}$ of $\mathcal{F}$, i.e. $\mathcal{F} \vDash c$, clause $p \subseteq c$ is a prime implicate of $\mathcal{F}$ iff $\mathcal{F} \vDash p$ and $\forall_{q \subsetneq p}, \mathcal{F} \nvDash q$. FPIc is the function problem of computing a prime implicate of $\mathcal{F}$ given an implicate c of $\mathcal{F}$.

Example 6. Lef $\mathcal{F} = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (\neg x_2 \vee \neg x_3)\}$. Given the implicant $t = (x_1 \wedge \neg x_2 \wedge \neg x_3)$, the terms $u_1 = (x_1 \wedge \neg x_2)$ and $u_2 = (x_1 \wedge \neg x_3)$ are prime implicants of $\mathcal{F}$. Given the implicate $c = (x_1 \vee \neg x_3)$, the clause $p = (x_1)$ is a prime implicate of $\mathcal{F}$.

Observe that computing a prime implicant from a given implicant for a formula in CNF can be done in polynomial time (e.g. [128,39]). Similarly, computing a prime implicate from a given implicate for a formula in DNF can also be done in polynomial time.

For the general case of arbitrary Boolean formulae these function problems become significantly harder, and the algorithms described in this paper require a number of SAT solver calls that is linear in the number of variables in the worst case (see [33,147] for similar approaches and conclusions).

Prime implicates and implicants find many practical applications, including truth maintenance systems [130,77], knowledge compilation [38,124], expert systems [144], conformant planning [145], the simplification of Boolean functions [127,112,64], abstraction in model checking [26,25], minimization of counterexamples in model checking [128,137,136], among many others. A wealth of algorithms exist for computing prime implicates and implicants (e.g. [23,30,111,66,124] and references therein).

Another problem of interest in practice is the Longest Extension of Implicant problem [147] for DNF formulae. By analogy, we also consider the Longest Extension of Implicate problem for CNF formulae.

Definition 26 *(Longest Extension of Implicant; FLEIt).* Let t be an implicant of $\mathcal{F} \in \mathbb{D}$, i.e. $t \vDash \mathcal{F}$. Term u, with $\mathbb{A} \supseteq u \supseteq t$, is a *longest extension of implicant* t iff $(\mathcal{F} \setminus \{t\}) \cup \{u\} \equiv \mathcal{F}$ and $\forall_{u' \supsetneq u}, (\mathcal{F} \setminus \{t\}) \cup \{u'\} \not\equiv \mathcal{F}$. The FLEIt function problem consists of computing a longest extension of implicant t of $\mathcal{F}$.

Definition 27 *(Longest Extension of Implicate; FLEIc).* Let c be an implicate of $\mathcal{F} \in \mathbb{C}$, i.e. $\mathcal{F} \vDash c$. Clause u, with $\mathbb{A} \supseteq u \supseteq c$, is a *longest extension of implicate* c iff $(\mathcal{F} \setminus \{c\}) \cup \{u\} \equiv \mathcal{F}$ and $\forall_{u' \supsetneq u}, (\mathcal{F} \setminus \{c\}) \cup \{u\} \not\equiv \mathcal{F}$. The FLEIc function problem consists of computing a longest extension of implicate c of $\mathcal{F}$.

3.1.6. Formula entailment

A number of problems related with formula entailment can also be considered. We address two function problems: computing a minimal set entailing a formula and computing a maximal set entailed by a formula.

Definition 28 *(Minimal Entailing Subset; FMnES).* Let $\mathcal{J} \in \mathbb{C}$ and $\mathcal{I} \in \mathbb{F}$ be such that $\mathcal{J} \vDash \mathcal{I}$. $\mathcal{M} \subseteq \mathcal{J}$ is a *Minimal Entailing Subset* (MnES) of $\mathcal{I}$ iff $\mathcal{M} \vDash \mathcal{I}$ and $\forall_{\mathcal{M}' \subsetneq \mathcal{M}} \mathcal{M}' \nvDash \mathcal{I}$. FMnES is the function problem of computing a minimal entailing subset of $\mathcal{J}$ given $\mathcal{I}$.

Example 7. Let $\mathcal{J} = \{(x_1 \vee x_2), (\neg x_2 \vee \neg x_3), (x_3 \vee x_4)\}$ and $\mathcal{I} = \{(x_1 \vee \neg x_3)\}$. The formula $\mathcal{M} = \{(x_1 \vee x_2), (\neg x_2 \vee \neg x_3)\}$ is an MnES of $\mathcal{I}$.

Observation 3. The function problem FMnES from Definition 28 can be generalized to capture the case of logic-based abduction [46] when the universe of hypotheses is consistent with the theory, and the minimality criterion is subset minimality.

Observation 4. Computing minimal entailing subsets also finds application in abstract argumentation [150]. This becomes evident when we allow $\mathcal{J}$ to be a conjunction of propositional formulae. Another example of the application of minimal sets in argumentation is [17].

Definition 29 *(Maximal Entailed Subset; FMxES).* Let $\mathcal{J} \in \mathbb{F}$ and $\mathcal{N} \in \mathbb{C}$ be such that $\mathcal{J} \nvDash \mathcal{N}$. $\mathcal{I} \subseteq \mathcal{N}$ is a *Maximal Entailed Subset* (MxES) iff $\mathcal{J} \vDash \mathcal{I}$ and $\forall_{\mathcal{N} \supseteq \mathcal{I}' \supsetneq \mathcal{I}} \mathcal{J} \nvDash \mathcal{I}'$. FMxES is the function problem of computing a maximal entailed subset of $\mathcal{N}$ given $\mathcal{J}$.

Example 8. Let $\mathcal{J} = \{(x_1 \vee x_2), (\neg x_2 \vee x_3), (\neg x_3 \vee x_4)\}$ and $\mathcal{N} = \{(x_1 \vee x_3), (x_1 \vee x_4), (x_2)\}$. The formula $\mathcal{I} = \{(x_1 \vee x_3), (x_1 \vee x_4)\}$ is a MxES of $\mathcal{J}$.

3.1.7. Backbone literals

The problem of computing the backbone of a Boolean formula, i.e. the literals that are common to all satisfying assignments of the formula, finds a wide range of applications. Two definitions are considered.

Definition 30 *(Backbone; FBB).* Given $\mathcal{F} \in \mathbb{F}$, the *backbone* of $\mathcal{F}$ is a maximal set of literals $\mathcal{B} \in \mathbb{A}$ which are true in all models of $\mathcal{F}$, i.e. $\mathcal{F} \vDash \wedge_{l \in \mathcal{B}}(l)$. FBB is the function problem of computing the backbone of $\mathcal{F} \in \mathbb{F}$.

Example 9. Let $\mathcal{F} = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (\neg x_1 \vee \neg x_3)\}$. The backbone of $\mathcal{F}$ is the set of literals $\mathcal{B} = \{x_1, \neg x_3\}$.

In practice, algorithms for computing the backbone of a formula often start from a reference model [108,69]. This allows a slightly different formulation of the backbone function problem.

Definition 31 *(FBBr).* Let ν be a model of $\mathcal{F} \in \mathbb{F}$ and $\mathcal{V}$ the associated set of literals. The *backbone* of $\mathcal{F}$ is a maximal set of literals $\mathcal{B} \subseteq \mathcal{V}$ such that $\mathcal{F} \vDash \wedge_{l \in \mathcal{B}}(l)$. FBBr is the function problem of computing the backbone of $\mathcal{F} \in \mathbb{F}$ given $\mathcal{V}$.

Backbones of Boolean formulae were first studied in the context of work on phase transitions and problem hardness [115, 142,76]. In addition, backbones find several practical applications, that include configuration [138,139], abstract argumentation [151], cyclic causal models [59], debugging of fabricated circuits [156], among others. A number of recent algorithms have been proposed for computing the backbone of Boolean formulae [74,67,108,157,69]. Recently, generalized backbones were studied in [36].

For constraint networks, researchers have recently investigated *global inverse consistency* (GIC) [18], which can be viewed as a generalization of backbone literals for non-Boolean domains.

3.1.8. *Variable independence*

A formula $\mathcal{F}$ is (semantically) independent from a variable x if the set of models of $\mathcal{F}$ does not change by fixing x to *any* truth value [91, Def. 4, Prop. 7].

Definition 32 *(Variable Independence; FVInd).* A $\mathcal{F} \in \mathbb{F}$ is independent from $x \in \mathrm{var}(\mathcal{F})$ iff $\mathcal{F} \equiv \mathcal{F}_{x=0}$ (or, $\mathcal{F} \equiv \mathcal{F}_{x=1}$). FVInd is the function problem of computing a maximal set of variables of which $\mathcal{F}$ is independent from.

Example 10. Let $\mathcal{F} = \{(x_1 \vee x_2), (x_1 \vee \neg x_2)\}$. $\mathcal{F}$ is independent from variable x_1.

Observe that, as indicated earlier, the definition of MESes can be extended to variables. However, FVInd is defined in a more general setting, since $\mathcal{F}$ need not be in CNF. Variable independence (and also literal independence) are important in a number of settings [91]. Variables declared independent are also known as *redundant* or *inessential*, e.g. [41,28,37]. In later sections, and for simplicity, if $\mathcal{F}$ is independent from x, then we write that x is redundant for $\mathcal{F}$.

3.1.9. *Maximum autarkies*

This section provides a brief overview of the problem of identifying the (maximum) autarkies of unsatisfiable CNF formulae.

Definition 33 *(Autarky; FAut).* Given $\mathcal{F} \in \mathbb{C}$, with $\mathcal{F} \vDash \bot$, a set $\mathcal{A} \subseteq \mathrm{var}(\mathcal{F})$ is an autarky iff there exists a truth assignment to the variables in $\mathcal{A}$ that satisfies all clauses containing literals in the variables of $\mathcal{A}$. FAut is the function problem of computing the maximal autarky of $\mathcal{F}$.

Example 11. Let $\mathcal{F} = \{(\neg x_1), (x_1 \vee x_2), (x_1 \vee \neg x_2), (\neg x_2 \vee x_3), (\neg x_3 \vee x_4)\}$. The set of variables $\mathcal{A} = \{x_3, x_4\}$ is the maximal autarky of $\mathcal{F}$.

Autarkies were first proposed in the context of improving exponential upper bounds of SAT algorithms [116,149], and have later been studied in the context of minimal unsatisfiable subformulae [84,78,86]. More recently, autarkies were used in model elimination [148] and for speeding up MCS/MUS enumeration algorithms [100]. Algorithms for computing autarkies were studied in [85,88,100,78,104].

3.1.10. *Minimal independent support*

If a propositional formula $\mathcal{F}$ is defined on a set of variables X, then X is referred to as the support of $\mathcal{F}$. Let $\mathcal{I} \subseteq X$ be such that for every satisfying assignment of $\mathcal{F}$, the truth values assigned to the variables in $\mathcal{I}$ uniquely determine the values of $X \setminus \mathcal{I}$. Then, $\mathcal{I}$ is called and independent support of $\mathcal{F}$ and $D = X \setminus \mathcal{I}$ is called the dependent support [31,65]. A minimal independent support $\mathcal{I}$ is an independent support of $\mathcal{F}$ such that none of its proper subsets is an independent support of $\mathcal{F}$.

Definition 34 *(Minimal Independent Support; FMIS).* Let $\mathcal{F} \in \mathbb{F}$ with support X. $\mathcal{I} \subseteq X$ is a *Minimal Independent Support* (MIS) of $\mathcal{F}$ iff for any two satisfying assignments ν_1, ν_2 that agree on $\mathcal{I}$ then $\nu_1 = \nu_2$; and for all $\mathcal{I}' \subsetneq \mathcal{I}$ there exist two satisfying assignments ν_1', ν_2' that agree on $\mathcal{I}'$ and $\nu_1' \neq \nu_2'$. FMIS is the function problem of computing a minimal independent support of $\mathcal{F}$.

Example 12. Let $\mathcal{F} = \{(x_1 \vee \neg x_2) \vee (\neg x_1 \vee x_2)\}$. The sets of variables $\mathcal{I}_1 = \{x_1\}$ and $\mathcal{I}_2 = \{x_2\}$ are MISes of $\mathcal{F}$.

A minimum independent support is a minimal independent support of smallest cardinality. Minimal independent support was recently shown to find application in model sampling and counting [65].

3.2. Properties

This section summarizes properties of some of the function problems presented in the previous section, which are essential for some of the results presented later.

For some function problems, the number of maximal/minimal sets is very restricted. In fact, for FLEIt, FLEIc, FMxES, FBBr, FBB, FVInd and FAut, the following holds.

Proposition 5. *For the function problems FLEIt, FLEIc, FMxES, FBB, FBBr, FVInd and FAut there is a unique maximal set, which is maximum.*

Proof. For each case, the proof is by contradiction.

1. For FLEIt, let u be a subset-maximal extension of t such that $u \vDash \mathcal{F}$, and let l_x be a literal not included in u such that $t \wedge l_x \vDash \mathcal{F}$. Then, $u \wedge l_x \vDash \mathcal{F}$; a contradiction.
2. For FLEIc the proof is similar to the previous case. Let u be a subset-maximal extension of clause c such that $\mathcal{F} \vDash u$, and let l_x be a literal not included in u such that $\mathcal{F} \vDash (c \vee l_x)$. Then, $\mathcal{F} \vDash (u \vee l_x)$; a contradiction.
3. For FMxES, let $\mathcal{I}$ be a maximal set such that $\mathcal{J} \vDash \mathcal{I}$, and assume there exists clause $c \in \mathcal{N} \setminus \mathcal{I}$ such that $\mathcal{J} \vDash c$. Then, any model of $\mathcal{J}$ satisfies both $\mathcal{I}$ and c, and so $\mathcal{J} \vDash \mathcal{I} \cup \{c\}$; a contradiction.
4. For FBB/FBBr, the proof is similar. Let $\mathcal{B}$ denote a maximal set of backbone literals, and let $l \notin \mathcal{B}$ be a backbone literal. Then, for any model of the formula, all literals in $\mathcal{B} \cup \{l\}$ are true; a contradiction.
5. For FVInd, by definition, a formula $\mathcal{F}$ is independent from x_i iff $\mathcal{F} \equiv \mathcal{F}_{x_i=0} \equiv \mathcal{F}_{x_i=1}$. Thus, we can check each variable separately for independence. Let $\mathcal{I}$ be a maximal set of variables $\mathcal{F}$ is independent from, and assume there exists a variable x_j independent from $\mathcal{F}$ such that $x_j \notin \mathcal{I}$. Since x_j is independent from $\mathcal{F}$, then $\mathcal{F} \equiv \mathcal{F}_{x_j=0} \equiv \mathcal{F}_{x_j=1}$; a contradiction.
6. For FAut, the proof is again similar. Let $\mathcal{A}_1$ be a maximal autarky, let $\mathcal{A}_2$ be another autarky, and let $\mathcal{A}_2 \setminus \mathcal{A}_1 \neq \emptyset$. Then $\mathcal{K} = \mathcal{A}_1 \cup \mathcal{A}_2$ is an autarky and $\mathcal{A}_1 \subsetneq \mathcal{K}$; a contradiction.

Thus, for FLEIt, FLEIc, FMxES, FBB, FBBr, FVInd and FAut there is a unique maximal set which is maximum. □

The following result provides a compact definition for variable independence. It will be useful in the following section.

Proposition 6. *A formula $\mathcal{F}$ is independent from $x_i \in X$ iff $\mathcal{F} \equiv \mathcal{F}[x_i/y_i]$, where y_i is a new variable with $y_i \notin X$.*

Proof. By definition, $\mathcal{F}$ is independent from x_i iff $\mathcal{F} \equiv \mathcal{F}_{x_i=0}$ or $\mathcal{F} \equiv \mathcal{F}_{x_i=1}$. Hence, $\forall_{y_i \in \{0,1\}} \mathcal{F} \equiv \mathcal{F}[x_i/y_i]$, with $y_i \notin X$. Thus, $\mathcal{F} \equiv \mathcal{F}[x_i/y_i]$ with $y_i \notin X$. □

4. Reductions to MSMP

This section shows how each of the function problems defined in Section 3 can be represented as an instance of the MSMP problem. Section 4.1 introduces general predicate forms, which are shown to be monotone, and which simplify the presentation of the reductions in the Section 4.2. Section 4.2 is structured similarly to Section 3.1: (i) minimal unsatisfiability (and maximal satisfiability); (ii) irredundant subformulae; (iii) maximal falsifiability (and minimal satisfiability); (iv) minimal and maximal models; (v) prime implicates and implicants; (vi) backbone literals; (vii) formula entailment; (viii) variable independence; (ix) maximum autarkies; and (x) minimal independent support.

4.1. Predicate forms

In the next section, several function problems are reduced to the MSMP. The reduction involves specifying a reference set $\mathcal{R}$ and a monotone predicate P defined in terms of a set $\mathcal{W} \subseteq \mathcal{R}$. To simplify the description of the different monotone predicates, this section develops general predicate forms, which capture all of the monotone predicates developed in the next sections, and proves that all predicates of any of these forms are monotone. As a result, for any concrete predicate, monotonicity is an immediate consequence of the monotonicity of the general predicate forms.

Let element $u_i \in \mathcal{R}$ represent either a literal or a clause. Moreover, $\sigma(u_i)$ represents a Boolean formula built from u_i, where new variables may be used, but such that u_i is the only element from $\mathcal{R}$ used in $\sigma(u_i)$. For example, $\sigma(u_i)$ can represent the negation of a literal or a clause, etc. Let $\mathcal{G}$ be a propositional formula that is *independent* from the elements in $\mathcal{W}$, i.e. $\mathcal{G}$ does not change with $\mathcal{W}$. Then, the following general predicate forms are defined.

Definition 35 *(Predicates of Form $\mathscr{L}$).* A predicate P is of *form* $\mathscr{L}$ iff its general form is given by,

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{R} \setminus \mathcal{W}} (\sigma(u_i))) \tag{1}$$

Definition 36 *(Predicates of Form $\mathscr{P}$).* A predicate P is of *form* $\mathscr{P}$ iff its general form is given by,

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{W}} (\sigma(u_i))) \tag{2}$$

Definition 37 *(Predicates of Form $\mathscr{B}$).* A predicate P is of *form* $\mathscr{B}$ iff its general form is given by,

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{G} \wedge (\vee_{u_i \in \mathcal{R} \setminus \mathcal{W}} (\sigma(u_i)))) \tag{3}$$

Proposition 7. *Predicates of the forms $\mathscr{L}$, $\mathscr{B}$ and $\mathscr{P}$ are monotone.*

Proof.

1. Let P be a predicate of form $\mathscr{L}$. Let $P(\mathcal{R}_0)$ hold, with $\mathcal{R}_0 \subseteq \mathcal{R}$, i.e. the argument to the SAT oracle is satisfiable. Then, by Proposition 1, for any $\mathcal{R}_1$, with $\mathcal{R}_1 \supseteq \mathcal{R}_0$ (and so $\mathcal{R} \setminus \mathcal{R}_1 \subseteq \mathcal{R} \setminus \mathcal{R}_0$), $P(\mathcal{R}_1)$ also holds, since the argument to the SAT oracle call is the conjunction of a subset of the constraints used for the case of $\mathcal{R}_0$, and so also satisfiable. Thus, P is monotone.
2. Let P be a predicate of form $\mathscr{P}$. Let $P(\mathcal{R}_0)$ hold, with $\mathcal{R}_0 \subseteq \mathcal{R}$, i.e. the argument to the SAT oracle is unsatisfiable. Then, by Proposition 2, for any $\mathcal{R}_1$, with $\mathcal{R}_1 \supseteq \mathcal{R}_0$, $P(\mathcal{R}_1)$ also holds, since the argument to the SAT oracle call is the conjunction of a superset of the constraints used for the case of $\mathcal{R}_0$, and so also unsatisfiable. Thus, P is monotone.
3. Let P be a predicate of form $\mathscr{B}$. Let $P(\mathcal{R}_0)$ hold, with $\mathcal{R}_0 \subseteq \mathcal{R}$, i.e. the argument to the SAT oracle is unsatisfiable. Then, for any $\mathcal{R}_1$, with $\mathcal{R}_1 \supseteq \mathcal{R}_0$, $P(\mathcal{R}_1)$ also holds, since the clause created from $\mathcal{R} \setminus \mathcal{R}_1$ has fewer elements than for the case of $\mathcal{R} \setminus \mathcal{R}_0$, and so it is also unsatisfiable (i.e. all literals in the clause will also be resolved away). Thus, P is monotone. □

Observe that Proposition 7 provides sufficient conditions for monotonicity. As shown later, there are monotone predicates that are not represented in one of the proposed forms, which will be used in some reductions.

4.2. Minimal sets

In the remainder of this section the reference set is denoted $\mathcal{R}$ and the monotone predicate P is defined in terms of a set $\mathcal{W} \subseteq \mathcal{R}$. All function problems considered are defined in Section 3.

The proofs presented in this section follow a general structure: they first introduce a predicate that reduces the given problem to an instance of MSMP. Then, monotonicity is proved identifying the predicate as one of form $\mathscr{L}$, $\mathscr{P}$ or $\mathscr{B}$. Finally, the correctness of the reduction is proven, showing that a minimal set over the predicate, or its complement, represents a solution to the given problem.

4.2.1. Minimal unsatisfiability & maximal satisfiability

The following propositions consider the function problems of computing a Minimal Unsatisfiable Subset (FMUS) and a Minimal Correction Subset (FMCS) of an unsatisfiable formula $\mathcal{F}$ in CNF.

Proposition 8. FMUS $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathcal{F}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\wedge_{c \in \mathcal{W}} (c)) \tag{4}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (4)) is of form $\mathscr{P}$, with $\mathcal{G} \triangleq \emptyset$, $u_i \triangleq c$, and $\sigma(c) \triangleq c$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set for which $P(\mathcal{M})$ holds, i.e. $\mathcal{M} \vDash \perp$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\mathcal{M}' \nvDash \perp$. Thus, by Definition 5, $\mathcal{M}$ is an MUS of $\mathcal{F}$. □

Proposition 9. FMCS $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathcal{F}$ and

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(c)) \tag{5}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (5)) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \emptyset$, $u_i \triangleq c$, and $\sigma(c) \triangleq c$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F} \setminus \mathcal{M} \nvDash \bot$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\mathcal{F} \setminus \mathcal{M}' \vDash \bot$. Thus, by Definition 6, $\mathcal{M}$ is an MCS of $\mathcal{F}$. □

Remark 8. By Remark 1, an MSS of $\mathcal{F} \in \mathbb{C}$ can be computed as follows. Compute an MCS $\mathcal{M}$ of $\mathcal{F}$ and return $\mathcal{F} \setminus \mathcal{M}$.

MSSes with assumptive contexts Recent work showed that computing one AC-MSS (see Definition 10) can be reduced to a sequence of calls to an NP oracle and a call to an MSS extractor [16]. We adapt this reduction to obtain a reduction of AC-MSS to MSMP.

Proposition 10. AC − MCS $\leq_p$ MSMP.

Proof.
Reduction. Associate with each clause $c_i \in \mathcal{F}$ a relaxation variable r_i, and consider a relaxed version of $\mathcal{F}$, defined on $X \cup R$, with $X \triangleq \mathsf{var}(\mathcal{F})$, and where R denotes the relaxation variables:

$$\mathcal{F}^R \triangleq \cup_{c_i \in \mathcal{F}}(c_i \vee r_i) \tag{6}$$

Create one copy of $\mathcal{F}^R$ for each $\gamma_j \in \Gamma$, by replacing X with a new set of variables X_j, i.e. $\mathcal{F}^R[X/X_j]$. Conjoin each such copy with the corresponding γ_j, and create the formula:

$$\mathcal{F}^{\mathrm{AC}} \triangleq \bigcup_{\gamma_j \in \Gamma} \mathcal{F}^R[X/X_j] \wedge \gamma_j \tag{7}$$

Now, let $\mathcal{R} \triangleq R$, and let,

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\mathcal{F}^{\mathrm{AC}} \wedge \wedge_{r_i \in \mathcal{R}\setminus\mathcal{W}}(r_i)) \tag{8}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (8)) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \mathcal{F}^{\mathrm{AC}}$, $u_i \triangleq r_i$, and $\sigma(r_i) \triangleq r_i$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F}^{\mathrm{AC}} \wedge \wedge_{r_i \in \mathcal{R}\setminus\mathcal{M}}(r_i) \nvDash \bot$, and so $(\mathcal{F} \setminus \mathcal{M}_c) \wedge \{\gamma_j\} \nvDash \bot$ for all $\gamma_j \in \Gamma$, where $\mathcal{M}_c$ denotes the clauses in $\mathcal{F}$ corresponding to the relaxation variables in $\mathcal{M}$. Note that the r_i variables enable deciding which clauses of $\mathcal{F}$ to include. This is then reflected in the different copies of $\mathcal{F}$, obtained by using new sets of variables. However, since the r_i variables are shared, then the set of clauses considered for each copy of $\mathcal{F}$ is the same. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, and so we have that for all $r' \in \mathcal{M}$, $\mathcal{F}^{\mathrm{AC}} \wedge \wedge_{r_i \in \mathcal{R}\setminus(\mathcal{M}-\{r'\})}(r_i) \vDash \bot$, and thus for all $c' \in \mathcal{M}_c$, $(\mathcal{F} \setminus \mathcal{M}_c) \wedge \{c'\} \wedge \{\gamma_j\} \vDash \bot$ for some $\gamma_j \in \Gamma$. Thus, by Definition 10, $\mathcal{F} \setminus \mathcal{M}_c$ is an MSS of $\mathcal{F}$ under a set of assumptive contexts Γ. Its complement, $\mathcal{M}_c$ represents an AC-MCS. □

Compared to earlier work [16], the proposed reduction trades off a possibly larger representation for a possible smaller number of oracle calls. Whereas earlier work [16] requires a number of NP oracle calls (linear on the number of assumptive formulae) to simplify the problem representation for extracting one MSS, our reduction only requires extracting one MCS (and so one MSS) at the expense of a possibly larger representation.

4.2.2. *Irredundant subformulae*

This section provides reductions for the function problems of computing a Minimal Equivalent Subset (FMES) and a Minimal Distinguishing Subset (FMDS) of a formula $\mathcal{F}$ in CNF.

Proposition 11. FMES $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathcal{F}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\neg\mathcal{F} \wedge \wedge_{c\in\mathcal{W}}(c)) \tag{9}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (9)) is of form $\mathscr{P}$, with $\mathcal{G} \triangleq \neg\mathcal{F}$, $u_i \triangleq c$, and $\sigma(c) \triangleq c$. Thus, by Proposition 7 the predicate is monotone.

Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{M} \vDash \mathcal{F}$. Also, as $\mathcal{F} \in \mathbb{C}$ and $\mathcal{M} \subseteq \mathcal{F}$, then $\mathcal{F} \vDash \mathcal{M}$. So, $\mathcal{F} \equiv \mathcal{M}$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\mathcal{M}' \nvDash \mathcal{F}$ (thus $\mathcal{F} \not\equiv \mathcal{M}'$). Thus, by Definition 11, $\mathcal{M}$ is an MES of $\mathcal{F}$. $\square$

Observation 5. Observe that the problem of computing an irredundant subformula can be reduced to the problem of computing a (group) MUS [9]. Thus, an MUS for the resulting problem is an MES for the original problem.

Also, note that the argument of the predicate in (9) can be simplified:

$$\begin{aligned}\neg\mathcal{F} \wedge \wedge_{c\in\mathcal{W}}(c) &\Leftrightarrow (\vee_{c\in\mathcal{F}}(\neg c)) \wedge \wedge_{c\in\mathcal{W}}(c) \Leftrightarrow (\vee_{c\in\mathcal{F}\setminus\mathcal{W}}(\neg c)) \wedge \wedge_{c\in\mathcal{W}}(c)\\ &\Leftrightarrow \neg(\mathcal{F}\setminus\mathcal{W}) \wedge \wedge_{c\in\mathcal{W}}(c)\end{aligned}$$

Thus, the predicate in (9) can be formulated as follows:

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\neg(\mathcal{F}\setminus\mathcal{W}) \wedge \wedge_{c\in\mathcal{W}}(c)) \tag{10}$$

Observe that the reduction above that does not respect the monotonicity conditions outlined earlier. However, the proposed predicate is monotone. As highlighted earlier, Proposition 7 represents sufficient conditions for monotonicity, and there are monotone predicates not represented in one of the proposed forms $\mathscr{P}$, $\mathscr{L}$ or $\mathscr{B}$.

Throughout the paper, (9) is used, since it facilitates relating this predicate with others. However, for practical purposes (10) would be preferred.

Proposition 12. FMDS $\le_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathcal{F}$ and

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\neg\mathcal{F} \wedge \wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(c)) \tag{11}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (11)) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \neg\mathcal{F}$, $u_i \triangleq c$, and $\sigma(c) \triangleq c$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{D}$ be a minimal set such that $P(\mathcal{D})$ holds, i.e. $\neg\mathcal{F} \wedge (\mathcal{F}\setminus\mathcal{D}) \nvDash \bot$. Thus $\mathcal{F}\setminus\mathcal{D} \nvDash \mathcal{F}$, and so $\mathcal{F} \not\equiv \mathcal{F}\setminus\mathcal{D}$. By Definition 2, since $\mathcal{D}$ is minimal for P, then for any $\mathcal{D}' \subsetneq \mathcal{D}$ the predicate does not hold, i.e. $\mathcal{F}\setminus\mathcal{D}' \vDash \mathcal{F}$. At the same time, as $\mathcal{F} \in \mathbb{C}$ and $\mathcal{F}\setminus\mathcal{D}' \subseteq \mathcal{F}$, then $\mathcal{F} \vDash \mathcal{F}\setminus\mathcal{D}'$, and so $\mathcal{F} \equiv \mathcal{F}\setminus\mathcal{D}'$. Thus, by Definition 12, $\mathcal{D}$ is an MDS of $\mathcal{F}$. $\square$

Note that the proof of Proposition 12 makes use of the following equivalence, given two propositional formulae $\mathcal{F}_1$ and $\mathcal{F}_2$: $(\mathcal{F}_1 \wedge \neg\mathcal{F}_2) \nvDash \bot$ iff $\mathcal{F}_1 \nvDash \mathcal{F}_2$.

Observation 6. The problem of computing an irredundant subformula can be reduced to the problem of computing a (group) MUS (see Observation 5 and [9]), and so an MCS for the resulting problem is an MDS for the original problem.

Similarly to the FMES case, the argument to the predicate in (11) can be simplified:

$$\begin{aligned}\neg\mathcal{F} \wedge \wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(c) &\Leftrightarrow (\vee_{c\in\mathcal{F}}(\neg c)) \wedge \wedge_{c\in\mathcal{F}\setminus\mathcal{W}}(c) \Leftrightarrow (\vee_{c\in\mathcal{W}}(\neg c)) \wedge \wedge_{c\in\mathcal{F}\setminus\mathcal{W}}(c)\\ &\Leftrightarrow \neg\mathcal{W} \wedge \wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(c)\end{aligned}$$

Thus, the predicate in (11) can be formulated as follows:

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\neg\mathcal{W} \wedge \wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(c)) \tag{12}$$

Observe that the reduction above that does not respect the monotonicity conditions outlined earlier. However, the proposed predicate is monotone. This represents another example illustrating that Proposition 7 represents sufficient conditions for monotonicity, and that there are monotone predicates not represented in one of the proposed forms $\mathscr{P}$, $\mathscr{L}$ or $\mathscr{B}$.

Throughout the paper, (11) is used, since it facilitates relating this predicate with others. However, for practical purposes (12) would be preferred.

Remark 9. By Remark 3, an MNS of $\mathcal{F} \in \mathbb{C}$ can be computed as follows. Compute an MDS $\mathcal{D}$ of $\mathcal{F}$ and return $\mathcal{F}\setminus\mathcal{D}$.

4.2.3. Minimal satisfiability & maximal falsifiability

This section provides a reduction for the function problem of computing a Minimal Correction (for Falsifiability) Subset (FMCFS) of a formula $\mathcal{F}$ in CNF.

Proposition 13. FMCFS $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathcal{F}$ and

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(\neg c)) \tag{13}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (13)) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \emptyset$, $u_i \triangleq c$, and $\sigma(c) \triangleq \neg c$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F} \setminus \mathcal{M}$ is all-falsifiable. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\mathcal{F} \setminus \mathcal{M}'$ is not all-falsifiable. Thus, by Definition 17, $\mathcal{M}$ is an MCFS of $\mathcal{F}$. □

It should be noted that given a CNF formula composed only of soft clauses, an FMCFS as formulated above can be computed in polynomial time, e.g. by using unit propagation or a greedy procedure. However, if the problem formulation contains hard clauses, then a SAT solver is necessary.

Remark 10. By Remark 5, an MFS of $\mathcal{F} \in \mathbb{C}$ can be computed as follows. Compute an MCFS $\mathcal{C}$ of $\mathcal{F}$ and return $\mathcal{F} \setminus \mathcal{C}$.

4.2.4. Minimal & maximal models

This section considers the computation of minimal models (FMnM) and maximal models (FMxM) of an arbitrary propositional formula $\mathcal{F}$.

Proposition 14. FMnM $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq X \triangleq \mathsf{var}(\mathcal{F})$ and

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\mathcal{F} \wedge \wedge_{x\in\mathcal{R}\setminus\mathcal{W}}(\neg x)) \tag{14}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (14)) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \mathcal{F}$, $u_i \triangleq x$, and $\sigma(x) \triangleq \neg x$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds. Note that $\mathcal{M}$ denotes the set of variables that *can* be assigned value 1 (since the other variables *must* be assigned value 0). As $P(\mathcal{M})$ holds, there exists $\mathcal{M}' \subseteq \mathcal{M}$ such that $\nu(\mathcal{M}', X) \vDash \mathcal{F}$, i.e. $\mathcal{M}'$ is a model of $\mathcal{F}$. As $\mathcal{M}$ is minimal for P, then $\mathcal{M}$ equals $\mathcal{M}'$, and thus $\nu(\mathcal{M}, X) \vDash \mathcal{F}$, i.e. $\mathcal{M}$ is a model of $\mathcal{F}$. Also, by Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}'' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\nu(\mathcal{M}'', X) \nvDash \mathcal{F}$. Thus, by Definition 21, $\mathcal{M}$ is a minimal model of $\mathcal{F}$. □

Proposition 15. FMxM $\leq_p$ MSMP.

Proof. Let $\mathcal{F}^C$ be constructed from $\mathcal{F}$ by flipping the polarity of all literals in $\mathcal{F}$, i.e. replace l with $\neg l$ for $l \in \{x, \neg x \mid x \in \mathsf{var}(\mathcal{F})\}$. Observe that, $\mathcal{F}$ and $\mathcal{F}^C$ have the same parse tree excluding the leaves. Now, compute a minimal model M^C for $\mathcal{F}^C$, e.g. using Proposition 14. Then, as shown next, a maximal model for $\mathcal{F}$ is given by $\mathsf{var}(\mathcal{F}) \setminus M^C$.

Let M^C be any model of $\mathcal{F}^C$, and let $\nu^C(\mathcal{M}^C, X)$ denote the associated truth assignment. Consider the truth assignment ν obtained by flipping the value of all variables, and let $\mathcal{M}$ denote the variables assigned value 1, i.e. $\mathcal{M} = X \setminus \mathcal{M}^C$. Clearly, $\mathcal{M}^C$ is minimal iff $\mathcal{M}$ is maximal. Moreover, let $\mathcal{F}$ be obtained from $\mathcal{F}^C$ by complementing all of its literals. Thus, the leaves of the parse tree of $\mathcal{F}^C$ are complemented and the values assigned to the leaves are also complemented. Now, recall that, with the exception of the leaves, both $\mathcal{F}$ and $\mathcal{F}^C$ have the same parse tree, and the leaves are assigned the same values in both cases. Thus, by structural induction it follows that $\nu^C \vDash \mathcal{F}^C$ iff $\nu \vDash \mathcal{F}$. □

Maximal models can also be captured by a specific monotone predicate.

Proposition 16. FMxM $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq X \triangleq \mathsf{var}(\mathcal{F})$ and

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\mathcal{F} \wedge \wedge_{x \in \mathcal{R} \setminus \mathcal{W}} (x)) \tag{15}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (15)) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \mathcal{F}$, $u_i \triangleq x$, and $\sigma(x) \triangleq x$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds. Note that $\mathcal{M}$ denotes the set of variables that *can* be assigned value 0 and $\mathcal{R} \setminus \mathcal{M}$ is a set of variables that *must* be assigned value 1. As $P(\mathcal{M})$ holds, there exists $\mathcal{M}' \subseteq \mathcal{M}$ such that $\nu(\mathcal{R} \setminus \mathcal{M}', X) \vDash \mathcal{F}$, i.e. $\mathcal{R} \setminus \mathcal{M}'$ is a model of $\mathcal{F}$. As $\mathcal{M}$ is minimal for P, then $\mathcal{M}$ equals $\mathcal{M}'$, and thus $\nu(\mathcal{R} \setminus \mathcal{M}, X) \vDash \mathcal{F}$, i.e. $\mathcal{R} \setminus \mathcal{M}$ is a model of $\mathcal{F}$. Also, by Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}'' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\nu(\mathcal{R} \setminus \mathcal{M}'', X) \nvDash \mathcal{F}$. As $\mathcal{M}$ is minimal $\mathcal{R} \setminus \mathcal{M}$ is maximal. Thus, by Definition 22, $\mathcal{R} \setminus \mathcal{M}$ is a maximal model of $\mathcal{F}$. □

Observation 7. Regarding the reduction in the proof of Proposition 16, note that a maximal model of $\mathcal{F}$ is given by $X \setminus \mathcal{M}$, with $\mathcal{M}$ a minimal set over the predicate (15).

4.2.5. Implicants & implicates

This section considers the function problems of computing a prime implicant given a term (FPIt), a prime implicate given a clause (FPIc), as well as the longest extension of an implicant (FLEIt) and the longest extension of an implicate (FLEIc). Recall that for FPIt and FPIc, $\mathcal{F}$ is an arbitrary propositional formula, whereas for FLEIt, $\mathcal{F}$ is in DNF and for FLEIc, $\mathcal{F}$ is in CNF.

Proposition 17. FPIt $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq L(t) \triangleq \{l \,|\, l \in t\}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\neg\mathcal{F} \wedge \wedge_{l \in \mathcal{W}} (l)) \tag{16}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (16)) is of form $\mathscr{P}$, with $\mathcal{G} \triangleq \neg\mathcal{F}$, $u_i \triangleq l$, and $\sigma(l) \triangleq l$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds. Thus, $\wedge_{l \in \mathcal{M}} (l) \vDash \mathcal{F}$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\wedge_{l \in \mathcal{M}'} (l) \nvDash \mathcal{F}$. Hence, as $\mathcal{M} \subseteq L(t)$, by Definition 24, the conjunction of the literals in $\mathcal{M}$ is a prime implicant of $\mathcal{F}$ given term t. □

Similarly, we can reduce the computation of a prime implicate given a clause to MSMP. (This reduction of FPIc to MSMP was first described in [25,26].)

Proposition 18. FPIc $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq L(c) \triangleq \{l \,|\, l \in c\}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F} \wedge \wedge_{l \in \mathcal{W}} (\neg l)) \tag{17}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (17)) is of form $\mathscr{P}$, with $\mathcal{G} \triangleq \mathcal{F}$, $u_i \triangleq l$, and $\sigma(l) \triangleq \neg l$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds. Thus, $\mathcal{F} \vDash (\vee_{l \in \mathcal{M}} l)$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\mathcal{F} \nvDash (\vee_{l \in \mathcal{M}'} l)$. Hence, as $\mathcal{M} \subseteq L(c)$, by Definition 25, the disjunction of the literals in $\mathcal{M}$ is a prime implicate of $\mathcal{F}$ given clause c. □

Regarding FLEIt, $\mathcal{F}$ is in DNF, $\mathcal{F} = \vee_{j=1}^{m} t_j$, and let the implicant to extend be t_k, with $1 \leq k \leq m$. Also, let define $\mathcal{F}^{\mathrm{ItX}} \triangleq (\mathcal{F}) \wedge \neg(\mathcal{F} \setminus \{t_k\})$, that is, $\mathcal{F}^{\mathrm{ItX}} \triangleq (\vee_{j=1}^{m} t_j) \wedge \neg((\vee_{j=1;j \neq k}^{m} t_j))$, which can be simplified to $\mathcal{F}^{\mathrm{ItX}} \triangleq t_k \wedge \wedge_{i=1,i \neq k}^{m} (\neg t_i)$.

Proposition 19. FLEIt $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathbb{L}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F}^{\mathrm{ItX}} \wedge (\vee_{l\in\mathcal{R}\setminus\mathcal{W}} \neg l)) \qquad (18)$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (18)) is of form $\mathscr{B}$, with $\mathcal{G} \triangleq \mathcal{F}^{\mathrm{ItX}}$, $u_i \triangleq l$, and $\sigma(l) \triangleq \neg l$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that P holds, and let $u = (\vee_{l\in\mathcal{R}\setminus\mathcal{M}} \neg l)$ be a clause and $q = \neg u$ be a term, i.e. $q = \wedge_{l\in\mathcal{R}\setminus\mathcal{M}}(l)$. As $P(\mathcal{M})$ holds, $\mathcal{F}^{\mathrm{ItX}} \wedge \{u\} \vDash \bot$, which is equivalent to $(\mathcal{F}) \wedge \neg(\mathcal{F} \setminus \{t_k\}) \wedge \{u\} \vDash \bot$, and so $\mathcal{F} \vDash (\mathcal{F} \setminus \{t_k\}) \vee q$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. for any $q' = \wedge_{l\in\mathcal{R}\setminus\mathcal{M}'} l$, we have $\mathcal{F} \nvDash (\mathcal{F} \setminus \{t_k\}) \vee q'$. Note $q' \supsetneq q$. Now we show that $t_k \subseteq q$. As $\mathcal{F}^{\mathrm{ItX}} \triangleq t_k \wedge \neg(\mathcal{F} \setminus \{t_k\})$, for any $l \in t_k$, we have $t_k \wedge \neg(\mathcal{F} \setminus \{t_k\}) \vDash l$, and so $\mathcal{F} \vDash (\mathcal{F} \setminus \{t_k\}) \vee l$. Suppose there is one $l \in t_k$ such that $l \notin q$, then we would have $\mathcal{F} \vDash (\mathcal{F} \setminus \{t_k\}) \vee (q \wedge l)$, thus contradicting the minimality of $\mathcal{M}$ (maximality of q). As $t_k \vDash \mathcal{F}$ and $(\mathcal{F} \setminus \{t_k\}) \vDash \mathcal{F}$ (recall $\mathcal{F} \in \mathbb{D}$) and $t_k \subseteq q$, then $(\mathcal{F} \setminus \{t_k\}) \vee q \vDash \mathcal{F}$. Hence, $\mathcal{F} \equiv (\mathcal{F} \setminus \{t_k\}) \vee q$ and, by Definition 26, q is the longest extension of implicant t_k of $\mathcal{F}$. Moreover, by Proposition 5, q is maximal and unique. □

Observation 8. Regarding the reduction in the proof of Proposition 19, note that the longest extension of implicant t_k of $\mathcal{F}$ is given by the conjunction of the literals in $\mathbb{L} \setminus \mathcal{M}$, being $\mathcal{M}$ a minimal set over the predicate (18).

Regarding FLEIc, $\mathcal{F}$ is in CNF, $\mathcal{F} = \wedge_{j=1}^{m}(c_j)$, and let the implicate to extend be c_k, with $1 \le k \le m$. Moreover, let define $\mathcal{F}^{\mathrm{IcX}} \triangleq (\neg\mathcal{F}) \wedge (\mathcal{F} \setminus \{c_k\})$, which can be simplified to $\mathcal{F}^{\mathrm{IcX}} \triangleq (\neg c_k) \wedge \wedge_{i=1, i\neq k}^{m}(c_i)$.

Proposition 20. FLEIc $\le_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathbb{L}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F}^{\mathrm{IcX}} \wedge (\vee_{l\in\mathcal{R}\setminus\mathcal{W}} l)) \qquad (19)$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (19)) is of form $\mathscr{B}$, with $\mathcal{G} \triangleq \mathcal{F}^{\mathrm{IcX}}$, $u_i \triangleq l$, and $\sigma(l) \triangleq l$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that P holds, and let $u = (\vee_{l\in\mathcal{R}\setminus\mathcal{M}} l)$. As $P(\mathcal{M})$ holds, $\mathcal{F}^{\mathrm{IcX}} \wedge \{u\} \vDash \bot$, which is equivalent to $(\neg\mathcal{F}) \wedge (\mathcal{F} \setminus \{c_k\}) \wedge \{u\} \vDash \bot$, and so $(\mathcal{F} \setminus \{c_k\}) \wedge \{u\} \vDash \mathcal{F}$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. for any $u' = (\vee_{l\in\mathcal{R}\setminus\mathcal{M}'} l)$, we have $(\mathcal{F} \setminus \{c_k\}) \wedge \{u'\} \nvDash \mathcal{F}$. Note $u' \supsetneq u$. Now we show that $c_k \subseteq u$. As $\mathcal{F}^{\mathrm{IcX}} \triangleq (\neg c_k) \wedge (\mathcal{F} \setminus \{c_k\})$, for any $l \in c_k$, we have $(\neg c_k) \wedge (\mathcal{F} \setminus \{c_k\}) \vDash \neg l$. Suppose there is one $l \in c_k$ such that $l \notin u$, then we would have $(\mathcal{F} \setminus \{c_k\}) \wedge (\{u\} \vee l) \vDash \mathcal{F}$, thus contradicting the minimality of $\mathcal{M}$ (maximality of u). As $\mathcal{F} \vDash \{c_k\}$ and $\mathcal{F} \vDash (\mathcal{F} \setminus \{c_k\})$ (recall $\mathcal{F} \in \mathbb{C}$) and $c_k \subseteq u$, then $\mathcal{F} \vDash (\mathcal{F} \setminus \{c_k\}) \wedge \{u\}$. Hence, $\mathcal{F} \equiv (\mathcal{F} \setminus \{c_k\}) \wedge \{u\}$ and, by Definition 27, u is the longest extension of implicate c_k of $\mathcal{F}$. Moreover, by Proposition 5, u is maximal and unique. □

Observation 9. Regarding the reduction in the proof of Proposition 20, note that the longest extension of clause c_k of $\mathcal{F}$ is given by the disjunction of the literals in $\mathbb{L} \setminus \mathcal{M}$, being $\mathcal{M}$ a minimal set over the predicate (19).

4.2.6. Formula entailment

The following propositions consider the function problems of computing a Minimal Entailing Subset (FMnES) and a Maximal Entailed Subset (FMxES). Recall that for FMnES, $\mathcal{I}$ is an arbitrary propositional formula and $\mathcal{J}$ is in CNF. For FMxES, $\mathcal{J}$ is an arbitrary propositional formula and $\mathcal{N}$ is in CNF.

Proposition 21. FMnES $\le_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathcal{J}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\neg\mathcal{I} \wedge \wedge_{c\in\mathcal{W}}(c)) \qquad (20)$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (20)) is of form $\mathscr{P}$, with $\mathcal{G} \triangleq \neg\mathcal{I}$, $u_i \triangleq c$, and $\sigma(c) \triangleq c$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{M} \vDash \mathcal{I}$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, that is, $\mathcal{M}' \nvDash \mathcal{I}$. Hence, by Definition 28, $\mathcal{M}$ is a minimal entailing subset of $\mathcal{J}$ given $\mathcal{I}$. □

Proposition 22. FMxES $\le_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq \mathcal{N}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{J} \wedge (\vee_{c\in\mathcal{R}\setminus\mathcal{W}} \neg c)) \tag{21}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (21)) is of form $\mathscr{B}$, with $\mathcal{G} \triangleq \mathcal{J}$, $u_i \triangleq c$, and $\sigma(c) \triangleq \neg c$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{J} \vDash (\mathcal{R} \setminus \mathcal{M})$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, that is, $\mathcal{J} \nvDash (\mathcal{R} \setminus \mathcal{M}')$. Hence, by Definition 29, $\mathcal{N} \setminus \mathcal{M}$ is a maximal entailed subset of $\mathcal{N}$ given $\mathcal{J}$. Moreover, by Proposition 5, $\mathcal{N} \setminus \mathcal{M}$ is maximal and unique. $\square$

Observation 10. Regarding the reduction in the proof of Proposition 22, note that the maximal entailed subset of $\mathcal{N}$ given $\mathcal{J}$ is given by $\mathcal{N} \setminus \mathcal{M}$, being $\mathcal{M}$ a minimal set over the predicate (21).

4.2.7. Backbone literals

This section provides reductions for the function problem of computing the backbone (FBB) of a an arbitrary propositional formula $\mathcal{F}$. Recall that FBBr refers to computing the backbone given a model.

Proposition 23. FBBr $\leq_p$ MSMP.

Proof.
Reduction. Consider a set of literals $\mathcal{V}$, obtained from an initial satisfying assignment ν. The reduction is defined as follows. $\mathcal{R} \triangleq \mathcal{V}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F} \wedge (\vee_{l\in\mathcal{R}\setminus\mathcal{W}} \neg l)) \tag{22}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (22)) is of form $\mathscr{B}$, with $\mathcal{G} \triangleq \mathcal{F}$, $u_i \triangleq l$, and $\sigma(l) \triangleq \neg l$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F} \vDash \wedge_{l\in\mathcal{R}\setminus\mathcal{M}} l$. So, for all $l \in \mathcal{R} \setminus \mathcal{M}$, we have $\mathcal{F} \vDash l$, i.e. l is a backbone literal of $\mathcal{F}$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold. Then, for each $l' \in \mathcal{M}$, $P(\mathcal{M} - \{l'\})$ does not hold, and so we have that $\mathcal{F} \nvDash \wedge_{l\in(\mathcal{R}\setminus\mathcal{M})\cup\{l'\}} l$. Thus, l' is not a backbone literal of $\mathcal{F}$. Thus, by Definition 31, $\mathcal{V} \setminus \mathcal{M}$ is a maximal set of backbone literals of $\mathcal{F}$. Moreover, by Proposition 5, $\mathcal{V} \setminus \mathcal{M}$ is maximal and unique. $\square$

Observation 11. Regarding the reduction in the proof Proposition 23, the computed minimal set $\mathcal{T}$ is the *complement* of the set of backbone literals, which is given by $\mathcal{V} \setminus \mathcal{T}$.

In practice, and for efficiency reasons, algorithms for computing the backbone of a Boolean formula start from a reference satisfying assignment (i.e. the FBBr function problem) [69]. However, the reduction of the general backbone computation problem to MSMP yields interesting insights into the worst-case number of SAT oracle queries needed to compute the backbone of a Boolean formula.

In this respect, there are two different alternatives: A predicate that computes the set of *backbone variables*, and a predicate that computes the set of backbone literals. In the first case, after computing the backbone variables, a call to a SAT solver is required in order to discover the polarities of the variables, i.e., the backbone literals. These reductions are illustrated in the two following propositions.

Proposition 24. FBB $\leq_p$ MSMP.

Proof.
Reduction. The reduction is defined as follows. $\mathcal{R} \triangleq X$, with $X = \mathsf{var}(\mathcal{F})$. Consider one auxiliary set of variables X', with $|X'| = |X|$, and let:

$$\mathcal{F}^{\mathrm{BB}} \triangleq \mathcal{F}[X/X] \wedge \mathcal{F}[X/X'] \tag{23}$$

Finally, let:

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F}^{\mathrm{BB}} \wedge (\vee_{x\in\mathcal{R}\setminus\mathcal{W}} x \wedge \neg x')) \tag{24}$$

Monotonicity. The predicate (see (24)) is of form $\mathscr{B}$, with $\mathcal{G} \triangleq \mathcal{F}^{\mathrm{BB}}$, $u_i \triangleq x$, and $\sigma(x) \triangleq x \wedge \neg x'$, where x' is a new variable not in $\mathcal{R}$, but associated with x. Thus, by Proposition 7 the predicate is monotone.

Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F}^{\text{BB}} \vDash \wedge_{x\in\mathcal{R}\setminus\mathcal{M}}(\neg x \vee x')$. So, for all $x \in \mathcal{R}\setminus\mathcal{M}$, we have that $\mathcal{F}^{\text{BB}} \vDash x$ or $\mathcal{F}^{\text{BB}} \vDash \neg x$, and thus $\mathcal{F} \vDash x$ or $\mathcal{F} \vDash \neg x$, i.e. x is a backbone variable of $\mathcal{F}$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold. Then, for each $y \in \mathcal{M}$, $P(\mathcal{M} - \{y\})$ does not hold, and so we have that $\mathcal{F} \nvDash y$ and $\mathcal{F} \nvDash \neg y$. Thus, y is not a backbone variable of $\mathcal{F}$. Thus, $\mathcal{V}\setminus\mathcal{M}$ is a maximal set of backbone variables of $\mathcal{F}$. Moreover, $\mathcal{V}\setminus\mathcal{M}$ is maximal and unique. □

Proposition 25. FBB $\leq_p$ MSMP.

Proof.
Reduction. Consider the set of literals $\mathbb{L}$ of $\mathcal{F}$, as the reference set. The reduction is defined as follows. $\mathcal{R} \triangleq \mathbb{L}$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F} \wedge (\vee_{l\in\mathcal{R}\setminus\mathcal{W}}\neg l)) \tag{25}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (25)) is of form $\mathscr{B}$, with $\mathcal{G} \triangleq \mathcal{F}$, $u_i \triangleq l$, and $\sigma(l) \triangleq \neg l$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F} \vDash \wedge_{l\in\mathcal{R}\setminus\mathcal{M}} l$. So, for all $l \in \mathcal{R}\setminus\mathcal{M}$, we have $\mathcal{F} \vDash l$, i.e. l is a backbone literal of $\mathcal{F}$. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold. Then, for each $l' \in \mathcal{M}$, $P(\mathcal{M} - \{l'\})$ does not hold, and so we have that $\mathcal{F} \nvDash \wedge_{l\in(\mathcal{R}\setminus\mathcal{M})\cup\{l'\}} l$. Thus, l' is not a backbone literal of $\mathcal{F}$. Thus, by Definition 31, $\mathbb{L}\setminus\mathcal{M}$ is a maximal set of backbone literals of $\mathcal{F}$. Moreover, by Proposition 5, $\mathbb{L}\setminus\mathcal{M}$ is maximal and unique. □

Because of the reduction proposed in the previous proof, the function problem that is solved is the identification of the set of backbone literals. In this case, it is not necessary to make an additional call to the SAT solver, such as in the reductions shown in Proposition 23, where the SAT call is made at the beginning to obtain a model, and in Proposition 24, where the *backbone variables* are identified and a SAT call is necessary at the end to discover the backbone literals (i.e. the polarities of these variables). Nevertheless, the size of the reference set is, in this case, twice the size in the two previous cases.

Observe that devising a predicate for generalized backbones [36] is straightforward; simply use (25) but let $\mathcal{R}$ capture the set of possible variable equivalences. Similarly, it is simple to reduce global inverse consistency [18] to MSMP. (The notation from [18] is assumed.) For simplicity, let N denote a constraint network and let $\mathcal{F} \triangleq \text{CNF}(N)$ denote the CNF encoding of N. Also, let $X = \{x_1, \ldots, x_n\}$ denote the set of variables of N, all with domain $D = \{d_1, \ldots, d_k\}$. Moreover, consider a direct encoding where Boolean variable y_{ij} is assigned value 1 iff variable x_i is assigned value v_j. Finally, let $\mathcal{R} \triangleq \{y_{ij}\}$. Given the above, the reduction is as follows:

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F} \wedge (\vee_{y_{ij}\in\mathcal{R}\setminus\mathcal{W}}\, y_{ij})) \tag{26}$$

with $\mathcal{W} \subseteq \mathcal{R}$. The single minimal set $\mathcal{M} \subseteq \mathcal{R}$ for which $P(\mathcal{M})$ holds contains all the assignments for which there is no satisfying assignment. More interestingly, and given the results in [106], this means that GIC can be solved with a logarithmic number of oracle queries. Concretely, [106] proves that predicates of form $\mathscr{B}$ can be solved with a logarithmic number of witness oracle queries. Since GIC is reduced to MSMP in form $\mathscr{B}$, then it can be solved with a logarithmic number of oracle queries.

4.2.8. *Variable independence*

The following reduction captures the function problem of computing the set of independent variables (FVInd) from an arbitrary propositional formula.

Proposition 26. FVInd $\leq_p$ MSMP.

Proof.
Reduction. Consider the original set of variables X and another set of variables Y, such that $|Y| = |X|$. $\mathcal{F}$ is to be checked for equivalence against $\mathcal{F}[X/Y]$, i.e. a copy of itself using new variables. An additional constraint is that some of these variables are equivalent. The non-equivalence between the two formulae is captured as follows:

$$\mathcal{F}^{\text{VInd}} \triangleq ((\mathcal{F}[X/Y] \wedge \neg\mathcal{F}[X/X]) \vee (\neg\mathcal{F}[X/Y] \wedge \mathcal{F}[X/X])) \tag{27}$$

Given $\mathcal{F}^{\text{VInd}}$, the reduction is defined as follows. $\mathcal{R} \triangleq X$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F}^{\text{VInd}} \wedge \wedge_{x_i\in\mathcal{W}}(x_i \leftrightarrow y_i)) \tag{28}$$

Monotonicity. The predicate (see (28)) is of form $\mathscr{P}$, with $\mathcal{G} \triangleq \mathcal{F}^{\text{VInd}}$, $u_i \triangleq x_i$, and $\sigma(x_i) \triangleq x_i \leftrightarrow y_i$, where y_i is a new variable not in $\mathcal{R}$, but associated with x_i. Thus, by Proposition 7 the predicate is monotone.

Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F}[X/X] \equiv \mathcal{F}[X/Y]$ whenever $\wedge_{x_i \in \mathcal{M}} (x_i \leftrightarrow y_i)$. By Proposition 6, the variables in $\mathcal{R} \setminus \mathcal{M}$ are redundant for $\mathcal{F}$, as $\mathcal{F}$ remains equivalent regardless of the value they take. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold. So, for each $x' \in \mathcal{M}$, $P(\mathcal{M} - \{x'\})$ does not hold, and so we have that $\mathcal{F}[X/X] \not\equiv \mathcal{F}[X/Y]$ whenever $\wedge_{x_i \in \mathcal{M}-\{x'\}} (x_i \leftrightarrow y_i)$. Thus, we can conclude x' is not redundant for $\mathcal{F}$. Hence, by Definition 32, $X \setminus \mathcal{M}$ is a maximal set of variables $\mathcal{F}$ is independent from. Moreover, by Proposition 5, $X \setminus \mathcal{M}$ is maximal and unique. $\square$

Observation 12. Regarding the reduction in the proof of Proposition 26, note that the set of independent variables of $\mathcal{F}$ is given by $X \setminus \mathcal{M}$, being $\mathcal{M}$ a minimal set over the predicate (28).

4.2.9. Maximum autarkies

Autarkies can be captured with two different monotone predicate forms. The first predicate is of form $\mathscr{L}$.

Proposition 27. FAut $\leq_p$ MSMP *(form $\mathscr{L}$).*

Proof.
Reduction. The reduction uses a simplified version of the model proposed in [100]. Consider a CNF formula $\mathcal{F}$ with a set of variables $X = \mathsf{var}(\mathcal{F})$. Create new sets of variables X^+, X^0, X^1, such that for $x \in X$, x^+ indicates whether a variable is selected, and x^1 and x^0 replace, respectively, the literals x and $\neg x$ in the clauses of $\mathcal{F}$. Thus, $\mathcal{F}$ is transformed into a new formula $\mathcal{F}^{0,1}$, where each literal in x is translated either into x^1 or x^0. The resulting CNF formula, $\mathcal{F}^{\text{Aut}}$, consists of CNF-encoding the following sets of constraints. For each $x \in X$, add to $\mathcal{F}^{\text{Aut}}$ (the clauses resulting from encoding) $x^1 \leftrightarrow x^+ \wedge x$ and $x^0 \leftrightarrow x^+ \wedge \neg x$. If a clause $c^{0,1} \in \mathcal{F}^{0,1}$ has a literal in x, then add to $\mathcal{F}^{\text{Aut}}$ the clause $(x^+ \to c^{0,1})$. So, the variable x^+ activates all $c^{0,1}$ that mention x and either x^0 or x^1 (depending on the value of x). Now, $\mathcal{R} \triangleq X^+$ and

$$P(\mathcal{W}) \triangleq \mathsf{SAT}(\mathcal{F}^{\text{Aut}} \wedge \wedge_{x^+ \in \mathcal{R} \setminus \mathcal{W}} (x^+)) \tag{29}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (29)) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \mathcal{F}^{\text{Aut}}$, $u_i \triangleq x^+$, and $\sigma(x^+) = x^+$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F}^{\text{Aut}} \wedge \wedge_{x^+ \in \mathcal{R} \setminus \mathcal{M}} (x^+)$ is satisfiable. So, by the construction of $\mathcal{F}^{\text{Aut}}$, there exists an assignment to the variables associated with $\mathcal{R} \setminus \mathcal{M}$ that satisfies the subformula of $\mathcal{F}$ comprising the clauses with some literal in the variables associated with those in $\mathcal{R} \setminus \mathcal{M}$. Hence, the set of variables associated with $\mathcal{R} \setminus \mathcal{M}$ is an autarky. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold. Then, the set of variables associated with those in $\mathcal{R} \setminus \mathcal{M}'$ is not an autarky. Thus, by Definition 33, $\mathcal{R} \setminus \mathcal{M}$ denotes a maximal autarky of $\mathcal{F}$. Moreover, by Proposition 5, the maximal autarky is unique and maximum. $\square$

Observation 13. Regarding the reduction in the proof Proposition 27, the computed minimal set $\mathcal{M}$ is the *complement* of the set of autark variables, which is given by $\mathsf{var}(\mathcal{F}) \setminus \mathcal{M}$.

The model used in the reduction of Proposition 27 is referred to as Γ_1 in [104]. The same paper proposes two additional and more compact models, derived from Γ_1, that should be preferred for practical purposes. An alternative predicate for FAut is of form $\mathscr{B}$, as shown next.

Proposition 28. FAut $\leq_p$ MSMP *(form $\mathscr{B}$).*

Proof.
Reduction. As before, the reduction uses a simplified version of the model proposed in [100] (see proof of Proposition 27). Given these definitions, $\mathcal{R} \triangleq X^+$ and

$$P(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{F}^{\text{Aut}} \wedge (\vee_{x^+ \in \mathcal{R} \setminus \mathcal{W}} x^+)) \tag{30}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (30)) is of form $\mathscr{B}$, with $\mathcal{G} \triangleq \mathcal{F}^{\text{Aut}}$, $u_i \triangleq x^+$, and $\sigma(x^+) = x^+$. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F}^{\text{Aut}} \vDash \wedge_{x^+ \in \mathcal{R} \setminus \mathcal{M}} (\neg x^+)$ and so, for all $x^+ \in \mathcal{R} \setminus \mathcal{M}$, $\mathcal{F}^{\text{Aut}} \vDash (\neg x^+)$. By the construction of $\mathcal{F}^{\text{Aut}}$, there is no assignment to the variables associated with $\mathcal{R} \setminus \mathcal{M}$ that satisfies the subformula of $\mathcal{F}$ comprising the clauses with some literal in the variable associated with $\mathcal{R} \setminus \mathcal{M}$. Hence, $\mathcal{R} \setminus \mathcal{M}$ denotes a set of non-autark variables. By Definition 2, since $\mathcal{M}$ is minimal for P, then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold. Then, for all $x' \in \mathcal{M}$, $\mathcal{F}^{\text{Aut}} \nvDash \wedge_{x^+ \in (\mathcal{R} \setminus \mathcal{M}) \cup \{x'\}} (\neg x^+)$, and so, x' is an autark variable. Thus, $\mathcal{R} \setminus \mathcal{M}$ denotes a maximal set of non-autark variables. By Proposition 5, there is a unique maximal set of autark variables. From this, we can also conclude the uniqueness of the maximal set of non-autark variables ($\mathcal{R} \setminus \mathcal{M}$). So, by Definition 33, $\mathcal{M}$ denotes the maximal autarky of $\mathcal{F}$. $\square$

4.2.10. Minimal independent support

Recent work showed that computing one MIS can be reduced to group MUS [65]. We follow a similar approach to reduce FMIS to MSMP.

Proposition 29. FMIS $\le_p$ MSMP.

Proof.
Reduction. Associate with each variable in X an integer index. Let I_X be the set of integer indices of the variables in X. Two additional sets of variables are considered, Y and Z, and let the same indices be used. Moreover let,

$$\mathcal{F}^{\text{MIS}} \triangleq \mathcal{F}[X/Y] \wedge \mathcal{F}[X/Z] \wedge (\vee_{i \in I_X} \neg(y_i \leftrightarrow z_i)) \tag{31}$$

$\mathcal{F}^{\text{MIS}}$ holds if there are two satisfying assignments of $\mathcal{F}$ that differ in some variable. Define $\mathcal{R} \triangleq I_X$, and let,

$$P(\mathcal{W}) \triangleq \neg\text{SAT}(\mathcal{F}^{\text{MIS}} \wedge \wedge_{i \in \mathcal{W}} (y_i \leftrightarrow z_i)) \tag{32}$$

with $\mathcal{W} \subseteq \mathcal{R}$.
Monotonicity. The predicate (see (32)) is of form $\mathscr{P}$, with $\mathcal{G} \triangleq \mathcal{F}^{\text{MIS}}$, $u_i \triangleq i$, and $\sigma(i) \triangleq y_i \leftrightarrow z_i$ where y_i and z_i are new variables not in $\mathcal{R}$, but associated with the indices in i. Thus, by Proposition 7 the predicate is monotone.
Correctness. Let $\mathcal{M}$ be a minimal set such that $P(\mathcal{M})$ holds, i.e. $\mathcal{F}^{\text{MIS}} \wedge \wedge_{i \in \mathcal{M}} (y_i \leftrightarrow z_i) \vDash \bot$, and so there are no two satisfying assignments of $\mathcal{F}$ that agree on the variables with indices in $\mathcal{M}$ and disagree on some other variables. Thus, the variables with indices in $\mathcal{M}$ are an independent support of $\mathcal{F}$. By Definition 2, since $\mathcal{M}$ is minimal for P', then for any $\mathcal{M}' \subsetneq \mathcal{M}$ the predicate does not hold, i.e. $\mathcal{F}^{\text{MIS}} \wedge \wedge_{i \in \mathcal{M}'} (y_i \leftrightarrow z_i) \nvDash \bot$, and so the variables with indices in $\mathcal{M}'$ are not an independent support of $\mathcal{F}$. Thus, by Definition 34, the variables with indices in $\mathcal{M}$ are a minimal independent support of $\mathcal{F}$. $\square$

In contrast with FVInd, and as pointed out in earlier work [65], there can be multiple solutions of FMIS.

4.3. Optimization problems

To illustrate the modeling flexibility of monotone predicates, this section investigates how to solve optimization problems, namely computing a smallest Minimal Correction Subset (FSMCS), a smallest Minimal Distinguishing Subset (FSMDS), a largest Maximal Falsifiable Subset (FLMFS) and a smallest minimal model (FSMnM). In contrast with the previous section, the objective here is not to develop efficient algorithms or insight, but to show other uses of monotone predicates. It should be noted that computing cardinality minimal sets for some of the other function problems, e.g. FMUS, FMES, FPIt, FPIc, etc., is significantly harder, since these function problems are in the second level of the polynomial hierarchy, e.g. [53,147].

In the remainder of this section, unweighted formulations are considered, i.e. each clause is soft with weight 1.

Proposition 30. FSMCS $\le_p$ MSMP.

Proof.
Reduction. Let $\mathcal{B} = \{0, 1, 2, \ldots, |\mathcal{F}|\}$ denote the possible numbers of clauses that are required to be satisfied. Furthermore, for each clause $c_i \in \mathcal{F}$, create a relaxed copy $(\neg p_i \vee c_i)$ and let $\mathcal{F}^R$ denote the CNF formula where each clause c_i is replaced by its relaxed version. Let $\mathcal{P}$ denote the set of selection variables. For each $b_j \in \mathcal{B}$ create the constraint $\sum_{p_i \in \mathcal{P}} p_i \ge b_j$. Each of these constraints defines a lower bound on the number of satisfied clauses. Define $\mathcal{R} \triangleq \mathcal{B}$. Moreover, given $\mathcal{W} \subseteq \mathcal{R}$ let,

$$Q(\mathcal{W}) \triangleq \bigwedge_{b_j \in \mathcal{R} \setminus \mathcal{W}} \left(\sum_{p_i \in \mathcal{P}} p_i \ge b_j \right) \tag{33}$$

Finally, the predicate is defined as follows:

$$P(\mathcal{W}) \triangleq \text{SAT}\left(\mathcal{F}^R \wedge Q(\mathcal{W})\right) \tag{34}$$

Note that the reduction is indeed polynomial.
Monotonicity. Given the definition of $Q(\mathcal{W})$ in (33), the predicate (34) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \mathcal{F}^R$, $u_i \triangleq b_j$, and $\sigma(b_j) \triangleq \left(\sum_{p_i \in \mathcal{P}} p_i \ge b_j\right)$. Thus, by Proposition 7 the predicate is monotone.
Correctness. The set of true p_i variables picks a subset $\mathcal{S}$ of the clauses in $\mathcal{F}$, which is to be checked for satisfiability. Given the formulation, it holds that a subset-minimal set must correspond to a cardinality minimal set. If, by removing from $\mathcal{W}$ the element associated with some value b_j satisfies the predicate, then removing from $\mathcal{W}$ any element associated with the b_k of value no greater than b_j also satisfies the predicate, and this holds with the same truth assignment, namely the same

set of selected clauses. Thus, any minimal set must exclude the element associated with the largest value b_j such that the predicate holds, and must also exclude any element associated with b_k of value no greater than b_j. Thus, the elements not removed from $\mathcal{W}$ are associated with b_k representing sizes of sets of clauses such that not all can be simultaneously satisfied. Hence, the minimal set represents a smallest MCS of $\mathcal{F}$. $\square$

Proposition 31. FSMDS $\leq_p$ MSMP.

Proof.
Reduction. The definitions of the reduction in the proof of Proposition 30 apply. As a result, the predicate is defined as follows:

$$P(\mathcal{W}) \triangleq \mathsf{SAT}\left(\neg\mathcal{F} \wedge \mathcal{F}^R \wedge Q(\mathcal{W})\right) \tag{35}$$

Monotonicity. Given the definition of $Q(\mathcal{W})$ in (33), the predicate (35) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \neg\mathcal{F} \wedge \mathcal{F}^R$, $u_i \triangleq b_j$, and $\sigma(b_j) \triangleq \left(\sum_{p_i \in \mathcal{P}} p_i \geq b_j\right)$. Thus, by Proposition 7 the predicate is monotone.
Correctness. The set of true p_i variables picks a subset $\mathcal{S}$ of the clauses in $\mathcal{F}$. Thus, it must hold that $\mathcal{F} \vDash \mathcal{S}$. Then, one wants to pick a largest subset $\mathcal{N}$ of $\mathcal{F}$ such that $\mathcal{N} \nvDash \mathcal{F}$. A minimal set $\mathcal{M} \subseteq \mathcal{R}$ over P corresponds to a maximal set $\mathcal{R} \setminus \mathcal{M}$, containing all the constraints $\left(\sum_{p_i \in \mathcal{P}} p_i \geq b_j\right)$ that can be satisfied. Hence, this gives a largest set $\mathcal{N}$ of selected clauses such that $\mathcal{N} \nvDash \mathcal{F}$, i.e. $\mathcal{N}$ constitutes, by Definition 14, a largest MNS of $\mathcal{F}$. Thus, its complement, $\mathcal{M}$, corresponds to a smallest MDS of $\mathcal{F}$. $\square$

Proposition 32. FSMCFS $\leq_p$ MSMP.

Proof.
Reduction. The elements to relax in this case are the complements of the clauses, which will be satisfied (so that the clauses are falsified). As a result, define $\mathcal{N}^R \triangleq \wedge_{c_i \in \mathcal{F}}(\neg p_i \vee \neg c_i)$. The predicate can then be defined as follows:

$$P(\mathcal{W}) \triangleq \mathsf{SAT}\left(\mathcal{N}^R \wedge Q(\mathcal{W})\right) \tag{36}$$

Monotonicity. Given the definition of $Q(\mathcal{W})$ in (33), the predicate (36) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \mathcal{N}^R$, $u_i \triangleq b_j$, and $\sigma(b_j) \triangleq \left(\sum_{p_i \in \mathcal{P}} p_i \geq b_j\right)$. Thus, by Proposition 7 the predicate is monotone.
Correctness. The set of true p_i variables picks a subset $\mathcal{S}$ of the clauses in $\mathcal{F}$. In this case, by the construction of $\mathcal{N}^R$, the predicate holds for the subset $\mathcal{S}$ iff $\neg\mathcal{S} \nvDash \bot$, i.e. iff $\mathcal{S}$ is all-falsifiable. A minimal set $\mathcal{M} \subseteq \mathcal{R}$ over P corresponds to a maximal set $\mathcal{R} \setminus \mathcal{M}$, containing all the constraints $\left(\sum_{p_i \in \mathcal{P}} p_i \geq b_j\right)$ which is all-falsifiable. Hence, by Definition 19, $\mathcal{R} \setminus \mathcal{M}$ constitutes a largest MFS of $\mathcal{F}$. Thus, $\mathcal{M}$ is a smallest MCFS of $\mathcal{F}$. $\square$

Proposition 33. FSMnM $\leq_p$ MSMP.

Proof.
Reduction. The elements to relax in this case are the variables assigned value true. As a result, define $\mathcal{V}^R \triangleq \wedge_{x_i \in \mathcal{V}}(\neg p_i \vee \neg x_i)$. The predicate can then be defined as follows:

$$P(\mathcal{W}) \triangleq \mathsf{SAT}\left(\mathcal{F} \wedge \mathcal{V}^R \wedge Q(\mathcal{W})\right) \tag{37}$$

Monotonicity. Given the definition of $Q(\mathcal{W})$ in (33), the predicate (37) is of form $\mathscr{L}$, with $\mathcal{G} \triangleq \mathcal{F} \wedge \mathcal{V}^R$, $u_i \triangleq b_j$, and $\sigma(b_j) \triangleq \left(\sum_{p_i \in \mathcal{P}} p_i \geq b_j\right)$. Thus, by Proposition 7 the predicate is monotone.
Correctness. The set of true p_i variables picks a subset $\mathcal{V}$ of the variables in $\mathcal{F}$ that must be assigned value 0. Then, one wants to pick a largest set of variables that must be assigned value 0. A minimal set $\mathcal{M} \subseteq \mathcal{R}$ over P corresponds to a maximal set $\mathcal{R} \setminus \mathcal{M}$, containing all the constraints $\left(\sum_{p_i \in \mathcal{P}} p_i \geq b_j\right)$ such that there exists a model of $\mathcal{F}$ assigning value 0 to all the variables in $\mathcal{R} \setminus \mathcal{M}$. This model is given by $\mathcal{M}$. Hence, $\mathcal{R} \setminus \mathcal{M}$ constitutes a largest set of variables assigned value 0 in a model of $\mathcal{F}$, and thus, by Definition 23, $\mathcal{M}$ is a smallest minimal model of $\mathcal{F}$. $\square$

4.4. Summary of reductions

Table 1 summarizes the reductions described in Section 4.2 and in Section 4.3. For each problem, the associated reference provides the complete details of the reduction to the MSMP problem. The first part denotes function problems where the goal is to compute a minimal set. The second part denotes function problems that represent optimization problems. Besides

Table 1
Overview of function problems and MSMP reductions. Additional problems include FMSS, FMNS, FMFS, FLMSS, FLMNS, FLMFS and FLMxM. Other function problems, with reductions to MSMP similar to the ones above, are covered in Section 3. For each MSMP reduction, the table indicates the function problem, the reference set $\mathcal{R}$, the predicate $P(\mathcal{W})$ used in the reduction, defined on subsets $\mathcal{W} \in \mathcal{R}$, the form of the predicate and a reference to the proposition where the reduction is introduced. Recall that $\mathcal{F}$ refers to a formula, X is the set of variables, $\mathbb{L}$ the set of literals, c a clause and l a literal. For further details and other, more specific notation, the reader is referred to Section 4.

Problem	$\mathcal{R}$	$P(\mathcal{W}), \mathcal{W} \subseteq \mathcal{R}$	Form	Reduction
FMUS	$\mathcal{F}$	$\neg\mathsf{SAT}(\wedge_{c\in\mathcal{W}}(c))$	$\mathscr{P}$	Proposition 8, p. 34
FMCS	$\mathcal{F}$	$\mathsf{SAT}(\wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(c))$	$\mathscr{L}$	Proposition 9, p. 34
FACMCS	R	$\mathsf{SAT}(\mathcal{F}^{\mathrm{AC}} \wedge \wedge_{r_i\in\mathcal{R}\setminus\mathcal{W}}(r_i))$	$\mathscr{L}$	Proposition 10, p. 35
FMES	$\mathcal{F}$	$\neg\mathsf{SAT}(\neg\mathcal{F} \wedge \wedge_{c\in\mathcal{W}}(c))$	$\mathscr{P}$	Proposition 11, p. 35
FMDS	$\mathcal{F}$	$\mathsf{SAT}(\neg\mathcal{F} \wedge \wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(c))$	$\mathscr{L}$	Proposition 12, p. 36
FMCFS	$\mathcal{F}$	$\mathsf{SAT}(\wedge_{c\in\mathcal{R}\setminus\mathcal{W}}(\neg c))$	$\mathscr{L}$	Proposition 13, p. 37
FMnM	X	$\mathsf{SAT}(\mathcal{F} \wedge \wedge_{x\in\mathcal{R}\setminus\mathcal{W}}(\neg x))$	$\mathscr{L}$	Proposition 14, p. 37
FMxM	X	$\mathsf{SAT}(\mathcal{F} \wedge \wedge_{x\in\mathcal{R}\setminus\mathcal{W}}(x))$	$\mathscr{L}$	Proposition 16, p. 37
FPIt	$L(t)$	$\neg\mathsf{SAT}(\neg\mathcal{F} \wedge \wedge_{l\in\mathcal{W}}(l))$	$\mathscr{P}$	Proposition 17, p. 38
FPIc	$L(c)$	$\neg\mathsf{SAT}(\mathcal{F} \wedge \wedge_{l\in\mathcal{W}}(\neg l))$	$\mathscr{P}$	Proposition 18, p. 38
FLEIt	$\mathbb{L}$	$\neg\mathsf{SAT}(\mathcal{F}^{\mathrm{ItX}} \wedge (\vee_{l\in\mathcal{R}\setminus\mathcal{W}} \neg l))$	$\mathscr{B}$	Proposition 19, p. 38
FLEIc	$\mathbb{L}$	$\neg\mathsf{SAT}(\mathcal{F}^{\mathrm{IcX}} \wedge (\vee_{l\in\mathcal{R}\setminus\mathcal{W}} l))$	$\mathscr{B}$	Proposition 20, p. 39
FMnES	$\mathcal{J}$	$\neg\mathsf{SAT}(\neg\mathcal{I} \wedge \wedge_{c\in\mathcal{W}}(c))$	$\mathscr{P}$	Proposition 21, p. 39
FMxES	$\mathcal{N}$	$\neg\mathsf{SAT}(\mathcal{J} \wedge (\vee_{c\in\mathcal{R}\setminus\mathcal{W}} \neg c))$	$\mathscr{B}$	Proposition 22, p. 39
FBBr	$\mathcal{V}$	$\neg\mathsf{SAT}(\mathcal{F} \wedge (\vee_{l\in\mathcal{R}\setminus\mathcal{W}} \neg l))$	$\mathscr{B}$	Proposition 23, p. 40
FBB	X	$\neg\mathsf{SAT}(\mathcal{F}^{\mathrm{BB}} \wedge (\vee_{x\in\mathcal{R}\setminus\mathcal{W}}\, x \wedge \neg x'))$	$\mathscr{B}$	Proposition 24, p. 40
FVInd	X	$\neg\mathsf{SAT}(\mathcal{F}^{\mathrm{VInd}} \wedge \wedge_{x_i\in\mathcal{W}}(x_i \leftrightarrow y_i))$	$\mathscr{P}$	Proposition 26, p. 41
FAut	X^+	$\mathsf{SAT}(\mathcal{F}^{\mathrm{Aut}} \wedge \wedge_{x^+\in\mathcal{R}\setminus\mathcal{W}}(x^+))$	$\mathscr{L}$	Proposition 27, p. 42
FAut	X^+	$\neg\mathsf{SAT}(\mathcal{F}^{\mathrm{Aut}} \wedge (\vee_{x^+\in\mathcal{R}\setminus\mathcal{W}}\, x^+))$	$\mathscr{B}$	Proposition 28, p. 42
FMIS	I_X	$\neg\mathsf{SAT}(\mathcal{F}^{\mathrm{MIS}} \wedge \wedge_{i\in\mathcal{W}}(y_i \leftrightarrow z_i))$	$\mathscr{P}$	Proposition 29, p. 43
FSMCS	$\mathcal{F}$	$\mathsf{SAT}\left(\mathcal{F}^R \wedge Q(\mathcal{W})\right)$	$\mathscr{L}$	Proposition 30, p. 43
FSMDS	$\mathcal{F}$	$\mathsf{SAT}\left(\neg\mathcal{F} \wedge \mathcal{F}^R \wedge Q(\mathcal{W})\right)$	$\mathscr{L}$	Proposition 31, p. 44
FSMCFS	$\mathcal{F}$	$\mathsf{SAT}\left(\mathcal{N}^R \wedge Q(\mathcal{W})\right)$	$\mathscr{L}$	Proposition 32, p. 44
FSMnM	X	$\mathsf{SAT}\left(\mathcal{F} \wedge \mathcal{V}^R \wedge Q(\mathcal{W})\right)$	$\mathscr{L}$	Proposition 33, p. 44

the function problems summarized in Table 1, Section 3.1 and also Section 4.2 indicate that several other problems related with Boolean formulae can also be mapped into the MSMP problem. These include variants of the presented problems when considering groups of clauses, variables, hard clauses, etc.

Finally, the GIC problem [18] in CSP (see (26) on page 41) serves to illustrate that MSMP also finds application in non-Boolean domains.

5. Relationships between predicate forms

This section establishes some relations among different predicates. The first part establishes hitting set duality between predicate forms $\mathscr{P}$ and $\mathscr{L}$. The second part investigates the relationship between predicate forms $\mathscr{L}$ and $\mathscr{B}$.

5.1. Hitting set duality between predicates of forms $\mathscr{L}$ and $\mathscr{P}$

Hitting set duality relationships can be traced to the work of Reiter [129], with additional results for propositional formulae over the years [21,5,99]. A recent treatment of duality is exemplified by the work of Slaney [141], which also addresses the issue of monotone predicates. This section shows that hitting set duality manifests itself in terms of the predicate forms considered, and so can be viewed as a refinement of recent results [141].

Consider arbitrary but fixed set $\mathcal{R}$, formula $\mathcal{G}$, elements u_i and function σ.

Let predicate $P_{\mathscr{L}}$ be defined as $P_{\mathscr{L}}(\mathcal{W}) \triangleq \mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i\in\mathcal{R}\setminus\mathcal{W}}(\sigma(u_i)))$ and let $M_{\mathscr{L}}(\mathcal{R})$ be the set of all minimal sets over $P_{\mathscr{L}}$. Moreover, let predicate $P_{\mathscr{P}}$ be defined as $P_{\mathscr{P}}(\mathcal{W}) \triangleq \neg\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i\in\mathcal{W}}(\sigma(u_i)))$ and let $M_{\mathscr{P}}(\mathcal{R})$ be the set of all minimal sets over $P_{\mathscr{P}}$. The following result holds (the proof follows a structure similar to proofs presented in previous works [129,21,5,99,141]).

Theorem 1. *Let $\mathcal{M} \subseteq \mathcal{R}$ and $\mathcal{U} \subseteq \mathcal{R}$. The following holds:*

1. *$\mathcal{M} \in M_{\mathscr{L}}(\mathcal{R})$ if and only if $\mathcal{M}$ is an irreducible hitting set of $M_{\mathscr{P}}(\mathcal{R})$*
2. *$\mathcal{U} \in M_{\mathscr{P}}(\mathcal{R})$ if and only if $\mathcal{U}$ is an irreducible hitting set of $M_{\mathscr{L}}(\mathcal{R})$*

Proof. Part 1, If: Let $\mathcal{M}$ be an irreducible hitting set of $M_{\mathscr{P}}(\mathcal{R})$, and let $\mathcal{S} = \mathcal{R} \setminus \mathcal{M}$. First, since $\mathcal{M}$ is a hitting set of $M_{\mathscr{P}}(\mathcal{R})$, $\mathcal{S}$ cannot contain a set in $M_{\mathscr{P}}(\mathcal{R})$, and so, $\neg P_{\mathscr{P}}(\mathcal{S})$, that is $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i\in\mathcal{S}}(\sigma(u_i)))$ holds, which is equivalent to $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{R} \setminus \mathcal{M}} (\sigma(u_i)))$. Thus $P_{\mathscr{L}}(\mathcal{M})$ holds. Since $\mathcal{M}$ is an irreducible hitting set of $M_{\mathscr{P}}(\mathcal{R})$, for any element $u \in \mathcal{M}$, there exists $\mathcal{U} \in M_{\mathscr{P}}(\mathcal{R})$ such that $\mathcal{M} \cap \mathcal{U} = \{u\}$. Hence, for any $u \in \mathcal{M}$, the set $\mathcal{S} \cup \{u\}$ includes some minimal set in $M_{\mathscr{P}}(\mathcal{R})$, and so $P_{\mathscr{P}}(\mathcal{S} \cup \{u\})$ holds, i.e. $\neg\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{S} \cup \{u\}} (\sigma(u_i)))$, which is equivalent to $\neg\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{R} \setminus (\mathcal{M} - \{u\})} (\sigma(u_i)))$. Thus $P_{\mathscr{L}}(\mathcal{M} - \{u\})$ does not hold. We conclude that $\mathcal{M}$ is a minimal set over $P_{\mathscr{L}}$.

Part 1, Only If: Let $\mathcal{M}$ be a minimal set over $P_{\mathscr{L}}$ and let $\mathcal{S} = \mathcal{R} \setminus \mathcal{M}$. Since $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{R} \setminus \mathcal{M}} (\sigma(u_i)))$ holds, $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{S}} (\sigma(u_i)))$ holds, and thus $\neg\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{S}} (\sigma(u_i)))$ does not hold, i.e., $P_{\mathscr{P}}(\mathcal{S})$ does not hold. Hence, for any $\mathcal{U} \in M_{\mathscr{P}}(\mathcal{R})$, $\mathcal{U} \setminus \mathcal{S} \neq \emptyset$ (otherwise $\mathcal{U} \subseteq \mathcal{S}$), and so $\mathcal{U} \cap \mathcal{M} \neq \emptyset$, that is, $\mathcal{M}$ is a hitting set of $M_{\mathscr{P}}(\mathcal{R})$. Now, since $\mathcal{M}$ is a minimal set over $P_{\mathscr{L}}$, for every element $u \in \mathcal{M}$, $P_{\mathscr{L}}(\mathcal{M} - \{u\})$ does not hold, i.e. $\neg\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{R} \setminus (\mathcal{M} - \{u\})} (\sigma(u_i)))$ holds, which is equivalent to $\neg\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{S} \cup \{u\}} (\sigma(u_i)))$. Thus $P_{\mathscr{P}}(\mathcal{S} \cup \{u\})$ holds. Thus, for every element $u \in \mathcal{M}$, there exists a minimal set $\mathcal{U}$ over $P_{\mathscr{P}}$ such that $\mathcal{M} \cap \mathcal{U} = \{u\}$. By definition, $\mathcal{M}$ is an irreducible hitting set of $M_{\mathscr{P}}(\mathcal{R})$.

Part 2, If: Let $\mathcal{U}$ be an irreducible hitting set of $M_{\mathscr{L}}(\mathcal{R})$. So, for any $\mathcal{M} \in M_{\mathscr{L}}(\mathcal{R})$, $\mathcal{U} \cap \mathcal{M} \neq \emptyset$. Since $\mathcal{M}$ is a minimal set such that $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{R} \setminus \mathcal{M}} (\sigma(u_i)))$ holds , $\mathcal{S} = \mathcal{R} \setminus \mathcal{M}$ is a maximal set such that $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{S}} (\sigma(u_i)))$ holds, or equivalently, such that $P_{\mathscr{P}}$ does not hold. As for no $\mathcal{S}$ we have $\mathcal{U} \subseteq \mathcal{S}$, we can conclude $P_{\mathscr{P}}(\mathcal{U})$ holds. Since $\mathcal{U}$ is irreducible, for every element $u \in \mathcal{U}$, there exists a set $\mathcal{M} \in M_{\mathscr{L}}(\mathcal{R})$ such that $\mathcal{U} \cap \mathcal{M} = \{u\}$. Thus, for every $u \in \mathcal{U}$, there exists $\mathcal{M}$ such that $\mathcal{U}' = \mathcal{U} \setminus \{u\} \subseteq \mathcal{R} \setminus \mathcal{M}$, and as $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{R} \setminus \mathcal{M}} (\sigma(u_i)))$ holds, $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{U}'} (\sigma(u_i)))$ holds as well. Thus, $P_{\mathscr{P}}(\mathcal{U}')$ does not hold. We conclude that $\mathcal{U}$ is a minimal set over $P_{\mathscr{P}}$.

Part 2, Only If: Let $\mathcal{U}$ be a minimal set over $P_{\mathscr{P}}$. Since $\neg\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{U}} (\sigma(u_i)))$ holds, $\mathcal{U}$ cannot be included in any maximal set $\mathcal{S}$ such that $P_{\mathscr{P}}(\mathcal{S})$ does not hold, i.e. $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{S}} (\sigma(u_i)))$, and so, for every set $\mathcal{M} = \mathcal{R} \setminus \mathcal{S} \in M_{\mathscr{L}}(\mathcal{R})$, we have $\mathcal{U} \cap \mathcal{M} \neq \emptyset$, i.e. $\mathcal{U}$ is a hitting set of $M_{\mathscr{L}}(\mathcal{R})$. Now, since $\mathcal{U}$ is a minimal set over $P_{\mathscr{P}}$, for any element $u \in \mathcal{U}$, $P_{\mathscr{P}}(\mathcal{U} \setminus \{u\})$ does not hold, and so the set $\mathcal{U} \setminus \{u\}$ is included in some maximal set $\mathcal{S}$ such that $\mathsf{SAT}(\mathcal{G} \wedge \wedge_{u_i \in \mathcal{S}} (\sigma(u_i)))$ holds, being $\mathcal{S} = \mathcal{R} \setminus \mathcal{M}$ for some minimal set $\mathcal{M}$ such that $P_{\mathscr{L}}(\mathcal{M})$ holds. Hence, for any element $u \in \mathcal{U}$ there exists a minimal set $\mathcal{M}$ over $P_{\mathscr{L}}$ such that $\mathcal{U} \cap \mathcal{M} = \{u\}$. So, by definition, $\mathcal{U}$ is a minimal hitting set of $M_{\mathscr{L}}(\mathcal{R})$. □

An immediate corollary of Theorem 1 is the well-known hitting set duality between MUSes and MCSes of propositional formulae [129]. It also captures the hitting set duality between MESes and MDSes of propositional formulae. In addition, the general hitting set duality between predicates of form $\mathscr{P}$ and predicates of form $\mathscr{L}$ could serve to obtain additional insights. For instance, exploiting hitting set duality is the basis for current state-of-the-art MUS enumeration algorithms [99,125,97,98,158] or prime implicants/implicates knowledge compilation methods [124], as well as for approaches aimed at solving hard optimization problems in the second level of the polynomial hierarchy, such as computing a cardinality minimal MUS [63] or formula simplification [64]. Thus, the general hitting set duality proved above can also serve to obtain new algorithms for other challenging enumeration or optimization problems.

5.2. *Equivalence between predicates of forms $\mathscr{B}$ and $\mathscr{L}$*

We have seen that there are some function problems, such as autarkies, that can be captured by different predicates of both forms $\mathscr{B}$ and $\mathscr{L}$. At this point, the reader may be wondering if there exists a general relationship between these two kinds of predicates regardless of the particular function problem they represent. We show that for any predicate of form $\mathscr{B}$ there exists an equivalent predicate of form $\mathscr{L}$, the size of the latter being bounded by a quadratic increment w.r.t. the former.

Theorem 2. *Given a predicate $P_{\mathscr{B}}$ of form $\mathscr{B}$, the problem of computing a minimal set over $P_{\mathscr{B}}$ can be reduced to the problem of computing a minimal set over $P_{\mathscr{L}}$ of form $\mathscr{L}$.*

Proof. This is an immediate consequence of the results in [106,70]. □

6. Practical impact

The development of MSMP as a framework for representing minimal set problems has impacted research in a number of related areas, both from a practical and from a theoretical perspective.

The most visible impact is arguably the new generation of algorithms for finding minimal correction subsets [103,114,113]. Some of these new algorithms for finding MCSes require fewer than linear number of calls to a SAT oracle, this while retaining the original problem variables. Recent work on finding minimal unsatisfiable subformulas has also been influenced by MSMP [4]. Similarly, work on finding minimal equivalent subformulas [10] has been directly influenced by MSMP.

Some recent algorithms for MaxSAT represent another relevant line of work, that was motivated by insights from work on MSMP [60].

The insights provided by MSMP also enable developing a new generation of algorithms for computing autarkies [104,89], again achieving worst-case number of oracle calls that significantly improve earlier results.

The ability to relate apparently different problems resulted in an ever-increasing number of techniques being proposed for solving problems in novel ways [103,10,69,114].

Investigation on approaches for reducing the worst-case number of oracle calls also enabled novel results for computing the backbone of a propositional formula as well as the for MUS extraction when the number of MUSes is known to be constant [106,70].

7. Conclusions & research directions

This paper extends recent work [25,26,107] on monotone predicates and shows that a large number of function problems defined on Boolean formulae can be reduced to computing a minimal set over a monotone predicate. Besides Boolean formulae, the paper argues that monotone predicates find application in more expressive domains, including ILP, SMT and CSP.

A number of research directions can be envisioned. A natural question is to identify other function problems that can be reduced to the MSMP problem. Algorithms for MSMP are described elsewhere [107]. Another natural research question is whether additional algorithms can be developed. In addition, most practical algorithms for solving minimal set problems exploit a number of practical optimizations to reduce the number of oracle queries [11,107,103]. Another natural research question is whether these practical optimizations can be used in the more general setting of MSMP. Finally, another line of research is to develop precise query complexity characterizations of instantiations of the MSMP, with [106,70] representing initial work. Concrete examples of interest include FMUS, FPIc, FPIt, FBB, FVInd, among others.

Acknowledgements

We are grateful to the anonymous reviewers for their insightful comments and suggestions. This work was influenced by discussions with colleagues at CASL/UCD and at IST/INESC-ID. This work was partially supported by SFI PI grant BEACON (09/IN.1/I2618), and by FCT grants ATTEST (CMU-PT/ELE/0009/2009), POLARIS (PTDC/EIA-CCO/123051/2010). Part of this work was done while the first author was affiliated with CASL/UCD and with IST/INESC-ID, and while the third author was a postdoctoral researcher at CASL/UCD. Carlos Mencía is supported by grant TIN2016-79190-R.

References

[1] C. Ansótegui, M.L. Bonet, J. Levy, SAT-based MaxSAT algorithms, Artif. Intell. 196 (2013) 77–105.
[2] A. Avidor, U. Zwick, Approximating MIN 2-SAT and MIN 3-SAT, Theory Comput. Syst. 38 (2005) 329–345.
[3] F. Bacchus, J. Davies, M. Tsimpoukelli, G. Katsirelos, Relaxation search: a simple way of managing optional clauses, in: [27], 2014, pp. 835–841.
[4] F. Bacchus, G. Katsirelos, Using minimal correction sets to more efficiently compute minimal unsatisfiable sets, in: D. Kroening, C.S. Pasareanu (Eds.), CAV, Springer, 2015, pp. 70–86.
[5] J. Bailey, P.J. Stuckey, Discovery of minimal unsatisfiable subsets of constraints using hitting set dualization, in: M.V. Hermenegildo, D. Cabeza (Eds.), PADL, Springer, 2005, pp. 174–186.
[6] R.R. Bakker, F. Dikker, F. Tempelman, P.M. Wognum, Diagnosing and solving over-determined constraint satisfaction problems, in: R. Bajcsy (Ed.), IJCAI, Morgan Kaufmann, 1993, pp. 276–281.
[7] S. Bayless, N. Bayless, H.H. Hoos, A.J. Hu, SAT modulo monotonic theories, in: [22], 2015, pp. 3702–3709.
[8] A. Belov, A. Ivrii, A. Matsliah, J. Marques-Silva, On efficient computation of variable MUSes, in: A. Cimatti, R. Sebastiani (Eds.), SAT, Springer, 2012, pp. 298–311.
[9] A. Belov, M. Janota, I. Lynce, J. Marques-Silva, On computing minimal equivalent subformulas, in: M. Milano (Ed.), CP, Springer, 2012, pp. 158–174.
[10] A. Belov, M. Janota, I. Lynce, J. Marques-Silva, Algorithms for computing minimal equivalent subformulas, Artif. Intell. 216 (2014) 309–326.
[11] A. Belov, I. Lynce, J. Marques-Silva, Towards efficient MUS extraction, AI Commun. 25 (2012) 97–116.
[12] A. Belov, J. Marques-Silva, Minimally unsatisfiable Boolean circuits, in: K.A. Sakallah, L. Simon (Eds.), SAT, Springer, 2011, pp. 145–158.
[13] R. Ben-Eliyahu, R. Dechter, On computing minimal models, Ann. Math. Artif. Intell. 18 (1996) 3–27.
[14] R. Ben-Eliyahu-Zohary, L. Palopoli, Reasoning with minimal models: efficient algorithms and applications, Artif. Intell. 96 (1997) 421–449.
[15] J.L. Bentley, A.C. Yao, An almost optimal algorithm for unbounded searching, Inf. Process. Lett. 5 (1976) 82–87.
[16] P. Besnard, É. Grégoire, J. Lagniez, On computing maximal subsets of clauses that must be satisfiable with possibly mutually-contradictory assumptive contexts in: [22], 2015, pp. 3710–3716.
[17] P. Besnard, É. Grégoire, C. Piette, B. Raddaoui, MUS-based generation of arguments and counter-arguments, in: IRI, IEEE Systems, Man, and Cybernetics Society, 2010, pp. 239–244.
[18] C. Bessiere, H. Fargier, C. Lecoutre, Global inverse consistency for interactive constraint satisfaction, in: C. Schulte (Ed.), CP, Springer, 2013, pp. 159–174.
[19] A. Biere, PicoSAT essentials, JSAT 4 (2008) 75–97.
[20] A. Biere, M. Heule, H. van Maarenaum, T. Walsh (Eds.), Handbook of Satisfiability, Frontiers in AI and Applications, vol. 185, IOS Press, 2009.
[21] E. Birnbaum, E.L. Lozinskii, Consistent subsets of inconsistent systems: structure and behaviour, J. Exp. Theor. Artif. Intell. 15 (2003) 25–46.
[22] B. Bonet, S. Koenig (Eds.), Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25–30, 2015, Austin, Texas, USA, AAAI Press, 2015.
[23] G. Bossu, P. Siegel, Saturation, nonmonotonic reasoning and the closed-world assumption, Artif. Intell. 25 (1985) 13–63.
[24] Y. Boufkhad, O. Roussel, Redundancy in random SAT formulas, in: H.A. Kautz, B.W. Porter (Eds.), AAAI/IAAI, AAAI Press/The MIT Press, 2000, pp. 273–278.
[25] A.R. Bradley, Z. Manna, Checking safety by inductive generalization of counterexamples to induction, in: FMCAD, IEEE Press, 2007, pp. 173–180.
[26] A.R. Bradley, Z. Manna, Property-directed incremental invariant generation, Form. Asp. Comput. 20 (2008) 379–405.
[27] C.E. Brodley, P. Stone (Eds.), Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27–31, 2014, Québec City, Québec, Canada, AAAI Press, 2014.
[28] F.M. Brown, Boolean Reasoning: The Logic of Boolean Equations, Springer, 1990.
[29] S.R. Buss, J. Krajíček, G. Takeuti, Provably total functions in the bounded arithmetic theories R_3^i, U_2^i, and V_2^i, in: P. Clote, J. Krajíček (Eds.), Arithmetic, Proof Theory, and Computational Complexity, OUP, 1995, pp. 116–161.
[30] T. Castell, Computation of prime implicates and prime implicants by a variant of the Davis and Putnam procedure, in: ICTAI, IEEE Computer Society, 1996, pp. 428–429.

[31] S. Chakraborty, K.S. Meel, M.Y. Vardi, Balancing scalability and uniformity in SAT witness generator, in: DAC, ACM, 2014, 60:1–60:6.
[32] Z.Z. Chen, S. Toda, The complexity of selecting maximal solutions, in: Structure in Complexity Theory Conference, IEEE Computer Society, 1993, pp. 313–325.
[33] Z.Z. Chen, S. Toda, The complexity of selecting maximal solutions, Inf. Comput. 119 (1995) 231–239.
[34] J.W. Chinneck, E.W. Dravnieks, Locating minimal infeasible constraint sets in linear programs, INFORMS J. Comput. 3 (1991) 157–168.
[35] A. Chmeiss, V. Krawczyk, L. Sais, Redundancy in CSPs, in: M. Ghallab, C.D. Spyropoulos, N. Fakotakis, N.M. Avouris (Eds.), ECAI, IOS Press, 2008, pp. 907–908.
[36] M. Codish, Y. Fekete, A. Metodi, Backbones for equality, in: V. Bertacco, A. Legay (Eds.), Haifa Verification Conference, Springer, 2013, pp. 1–14.
[37] Y. Crama, P.L. Hammer, Boolean Functions – Theory, Algorithms, and Applications, Encyclopedia of Mathematics and Its Applications, vol. 142, Cambridge University Press, 2011.
[38] A. Darwiche, P. Marquis, A knowledge compilation map, J. Artif. Intell. Res. 17 (2002) 229–264.
[39] D. Déharbe, P. Fontaine, D.L. Berre, B. Mazure, Computing prime implicants, in: FMCAD, IEEE, 2013, pp. 46–52.
[40] C. Desrosiers, P. Galinier, A. Hertz, S. Paroz, Using heuristics to find minimal unsatisfiable subformulas in satisfiability problems, J. Comb. Optim. 18 (2009) 124–150.
[41] D.L. Dietmeyer, P.R. Schneider, Identification of symmetry, redundancy and equivalence of Boolean functions, IEEE Trans. Electron. Comput. EC-16 (1967) 804–817.
[42] E.W. Dijkstra, The saddleback search, Epistle EWD 934, Department of Computer Sciences, The University of Texas at Austin, 1985.
[43] N. Eén, N. Sörensson, An extensible SAT-solver, in: E. Giunchiglia, A. Tacchella (Eds.), SAT, Springer, 2003, pp. 502–518.
[44] N. Eén, N. Sörensson, Translating pseudo-Boolean constraints into SAT, JSAT 2 (2006) 1–26.
[45] T. Eiter, G. Gottlob, Propositional circumscription and extended closed-world reasoning are Π_2^p-complete, Theor. Comput. Sci. 114 (1993) 231–245.
[46] T. Eiter, G. Gottlob, The complexity of logic-based abduction, J. ACM 42 (1995) 3–42.
[47] A. Felfernig, M. Schubert, C. Zehentner, An efficient diagnosis algorithm for inconsistent constraint sets, Artif. Intell. Eng. Des. Anal. Manuf. 26 (2012) 53–62.
[48] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, 1979.
[49] É. Grégoire, J. Lagniez, On anti-subsumptive knowledge enforcement, in: LPAR, 2015, pp. 48–62.
[50] É. Grégoire, J. Lagniez, B. Mazure, An experimentally efficient method for (MSS, CoMSS) partitioning, in: [27], 2014, pp. 2666–2673.
[51] É. Grégoire, J. Lagniez, B. Mazure, Multiple contraction through Partial-Max-SAT, in: ICTAI, IEEE Computer Society, 2014, pp. 321–327.
[52] É. Grégoire, B. Mazure, C. Piette, On approaches to explaining infeasibility of sets of Boolean clauses, in: ICTAI (1), IEEE Computer Society, 2008, pp. 74–83.
[53] A. Gupta, Learning Abstractions for Model Checking, Ph.D. thesis, Carnegie Mellon University, 2006.
[54] A. Gurfinkel, A. Belov, J. Marques-Silva, Synthesizing safe bit-precise invariants, in: E. Ábrahám, K. Havelund (Eds.), Tools and Algorithms for the Construction and Analysis of Systems – 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5–13, 2014. Proceedings, in: Lecture Notes in Computer Science, vol. 8413, Springer, 2014, pp. 93–108.
[55] F. Hemery, C. Lecoutre, L. Sais, F. Boussemart, Extracting MUCs from constraint networks, in: G. Brewka, S. Coradeschi, A. Perini, P. Traverso (Eds.), ECAI, IOS Press, 2006, pp. 113–117.
[56] F. Heras, A. Morgado, J. Marques-Silva, An empirical study of encodings for group MaxSAT, in: L. Kosseim, D. Inkpen (Eds.), Canadian Conference on AI, Springer, 2012, pp. 85–96.
[57] F. Heras, A. Morgado, J. Marques-Silva, MaxSAT-based encodings for group MaxSAT, AI Commun. 28 (2015) 195–214.
[58] M. Heule, S. Weaver (Eds.), SAT, Lecture Notes in Computer Science, vol. 9340, Springer, 2015.
[59] A. Hyttinen, P.O. Hoyer, F. Eberhardt, M. Järvisalo, Discovering cyclic causal models with latent variables: a general sat-based procedure, in: UAI, 2013.
[60] A. Ignatiev, A. Morgado, V.M. Manquinho, I. Lynce, J. Marques-Silva, Progression in maximum satisfiability, in: [134], 2014, pp. 453–458.
[61] A. Ignatiev, A. Morgado, J. Marques-Silva, On reducing maximum independent set to minimum satisfiability, in: C. Sinz, U. Egly (Eds.), SAT, Springer, 2014, pp. 103–120.
[62] A. Ignatiev, A. Morgado, J. Planes, J. Marques-Silva, Maximal falsifiability – definitions, algorithms, and applications, in: K.L. McMillan, A. Middeldorp, A. Voronkov (Eds.), LPAR, Springer, 2013, pp. 439–456.
[63] A. Ignatiev, A. Previti, M.H. Liffiton, J. Marques-Silva, Smallest MUS extraction with minimal hitting set dualization, in: CP, 2015, pp. 173–182.
[64] A. Ignatiev, A. Previti, J. Marques-Silva, SAT-based formula simplification, in: [58], 2015, pp. 287–298.
[65] A. Ivrii, S. Malik, K.S. Meel, M.Y. Vardi, On computing minimal independent support and its applications to sampling and counting, Constraints 21 (1) (2016) 41–58.
[66] S. Jabbour, J. Marques-Silva, L. Sais, Y. Salhi, Enumerating prime implicants of propositional formulae in conjunctive normal form, in: E. Fermé, J. Leite (Eds.), JELIA, Springer, 2014, pp. 152–165.
[67] M. Janota, Do SAT solvers make good configurators?, in: S. Thiel, K. Pohl (Eds.), SPLC (2), Lero Int. Science Centre, University of Limerick, Ireland, 2008, pp. 191–195.
[68] M. Janota, G. Botterweck, J. Marques-Silva, On lazy and eager interactive reconfiguration, in: Proceedings of the 8th International Workshop on Variability Modelling of Software-Intensive Systems, VaMoS, ACM, 2014, pp. 8:1–8:8.
[69] M. Janota, I. Lynce, J. Marques-Silva, Algorithms for computing backbones of propositional formulae, AI Commun. 28 (2015) 161–177.
[70] M. Janota, J. Marques-Silva, On the query complexity of selecting minimal sets for monotone predicates, Artif. Intell. 233 (2016) 73–83.
[71] M. Jose, R. Majumdar, Bug-assist: assisting fault localization in ANSI-C programs, in: G. Gopalakrishnan, S. Qadeer (Eds.), CAV, Springer, 2011, pp. 504–509.
[72] M. Jose, R. Majumdar, Cause clue clauses: error localization using maximum satisfiability, in: M.W. Hall, D.A. Padua (Eds.), PLDI, ACM, 2011, pp. 437–446.
[73] U. Junker, QuickXplain: preferred explanations and relaxations for over-constrained problems, in: D.L. McGuinness, G. Ferguson (Eds.), AAAI, AAAI Press/The MIT Press, 2004, pp. 167–172.
[74] A. Kaiser, W. Küchlin, Detecting inadmissible and necessary variables in large propositional formulae, in: Intl. Joint Conf. on Automated Reasoning (Short Papers), 2001.
[75] D.J. Kavvadias, M. Sideri, E.C. Stavropoulos, Generating all maximal models of a Boolean expression, Inf. Process. Lett. 74 (2000) 157–162.
[76] P. Kilby, J.K. Slaney, S. Thiébaux, T. Walsh, Backbones and backdoors in satisfiability, in: M.M. Veloso, S. Kambhampati (Eds.), AAAI, AAAI Press/The MIT Press, 2005, pp. 1368–1373.
[77] J. de Kleer, An improved incremental algorithm for generating prime implicates, in: W.R. Swartout (Ed.), AAAI, AAAI Press/The MIT Press, 1992, pp. 780–785.
[78] H. Kleine Büning, O. Kullmann, Minimal unsatisfiability and autarkies, in: [20], 2009, pp. 339–401.
[79] H. Kleine Büning, T. Letterman, Propositional Logic: Deduction and Algorithms, Cambridge University Press, 1999.
[80] R. Kohli, R. Krishnamurti, P. Mirchandani, The minimum satisfiability problem, SIAM J. Discrete Math. 7 (1994) 275–283.

[81] A. Komuravelli, A. Gurfinkel, S. Chaki, E.M. Clarke, Automatic abstraction in SMT-based unbounded software model checking, in: [135], 2013, pp. 846–862.
[82] M. Koshimura, H. Nabeshima, H. Fujita, R. Hasegawa, Minimal model generation with respect to an atom set, in: FTP, 2009, pp. 49–59.
[83] M.W. Krentel, The complexity of optimization problems, J. Comput. Syst. Sci. 36 (1988) 490–509.
[84] O. Kullmann, Investigations on autark assignments, Discrete Appl. Math. 107 (2000) 99–137.
[85] O. Kullmann, On the use of autarkies for satisfiability decision, Electron. Notes Discrete Math. 9 (2001) 231–253.
[86] O. Kullmann, Constraint satisfaction problems in clausal form I: autarkies and deficiency, Fundam. Inform. 109 (2011) 27–81.
[87] O. Kullmann, Constraint satisfaction problems in clausal form II: minimal unsatisfiability and conflict structure, Fundam. Inform. 109 (2011) 83–119.
[88] O. Kullmann, I. Lynce, J. Marques-Silva, Categorisation of clauses in conjunctive normal forms: minimally unsatisfiable sub-clause-sets and the lean kernel, in: A. Biere, C.P. Gomes (Eds.), SAT, Springer, 2006, pp. 22–35.
[89] O. Kullmann, J. Marques-Silva, Computing maximal autarkies with few and simple oracle queries, in: [58], 2015, pp. 138–155.
[90] P. Laborie, An optimal iterative algorithm for extracting MUCs in a black-box constraint network, in: [134], 2014, pp. 1051–1052.
[91] J. Lang, P. Liberatore, P. Marquis, Propositional independence: formula-variable independence and forgetting, J. Artif. Intell. Res. 18 (2003) 391–443.
[92] C.M. Li, F. Manyà, MaxSAT, hard and soft constraints, in: [20], 2009, pp. 613–631.
[93] C.M. Li, Z. Zhu, F. Manyà, L. Simon, Optimizing with minimum satisfiability, Artif. Intell. 190 (2012) 32–44.
[94] P. Liberatore, Redundancy in logic I: CNF propositional formulae, Artif. Intell. 163 (2005) 203–232.
[95] P. Liberatore, Redundancy in logic II: 2CNF and Horn propositional formulae, Artif. Intell. 172 (2008) 265–299.
[96] P. Liberatore, Redundancy in logic III: non-monotonic reasoning, Artif. Intell. 172 (2008) 1317–1359.
[97] M.H. Liffiton, A. Malik, Enumerating infeasibility: finding multiple MUSes quickly, in: C.P. Gomes, M. Sellmann (Eds.), CPAIOR, Springer, 2013, pp. 160–175.
[98] M.H. Liffiton, A. Previti, A. Malik, J. Marques-Silva, Fast, flexible MUS enumeration, Constraints 21 (2) (2016) 223–250.
[99] M.H. Liffiton, K.A. Sakallah, Algorithms for computing minimal unsatisfiable subsets of constraints, J. Autom. Reason. 40 (2008) 1–33.
[100] M.H. Liffiton, K.A. Sakallah, Searching for autarkies to trim unsatisfiable clause sets, in: H. Kleine Büning, X. Zhao (Eds.), SAT, Springer, 2008, pp. 182–195.
[101] M.V. Marathe, S.S. Ravi, On approximation algorithms for the minimum satisfiability problem, Inf. Process. Lett. 58 (1996) 23–29.
[102] J. Marques-Silva, Computing minimally unsatisfiable subformulas: state of the art and future directions, J. Mult.-Valued Log. Soft Comput. 19 (2012) 163–183.
[103] J. Marques-Silva, F. Heras, M. Janota, A. Previti, A. Belov, On computing minimal correction subsets, in: F. Rossi (Ed.), IJCAI, IJCAI/AAAI, 2013.
[104] J. Marques-Silva, A. Ignatiev, A. Morgado, V.M. Manquinho, I. Lynce, Efficient autarkies, in: [134], 2014, pp. 603–608.
[105] J. Marques-Silva, M. Janota, Computing minimal sets on propositional formulae I: problems & reductions, CoRR, arXiv:1402.3011, 2014.
[106] J. Marques-Silva, M. Janota, On the query complexity of selecting few minimal sets, Electron. Colloq. Comput. Complex. 21 (2014) 31.
[107] J. Marques-Silva, M. Janota, A. Belov, Minimal sets over monotone predicates in Boolean formulae in: [135], 2013, pp. 592–607.
[108] J. Marques-Silva, M. Janota, I. Lynce, On computing backbones of propositional theories, in: H. Coelho, R. Studer, M. Wooldridge (Eds.), ECAI, IOS Press, 2010, pp. 15–20.
[109] J. Marques-Silva, I. Lynce, On improving MUS extraction algorithms, in: K.A. Sakallah, L. Simon (Eds.), SAT, Springer, 2011, pp. 159–173.
[110] J. Marques-Silva, I. Lynce, S. Malik, Conflict-driven clause learning SAT solvers, in: [20], 2009, pp. 131–153.
[111] P. Marquis, Consequence finding algorithms, in: D.M. Gabbay, P. Smets (Eds.), Handbook of Defeasible Reasoning and Uncertainty Management Systems, vol. 5, Springer, 2000, pp. 41–145, Chap. 3.
[112] E. McCluskey, Minimization of Boolean functions, Bell Labs Tech. J. 35 (1956) 1417–1444.
[113] C. Mencía, A. Ignatiev, A. Previti, J. Marques-Silva, MCS extraction with sublinear oracle queries, in: N. Creignou, D.L. Berre (Eds.), Proceedings of the 19th International Conference on Theory and Applications of Satisfiability Testing, SAT 2016, Bordeaux, France, July 5–8, 2016, Springer, 2016, pp. 342–360.
[114] C. Mencía, A. Previti, J. Marques-Silva, Literal-based MCS extraction, in: [153], 2015, pp. 1973–1979.
[115] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, L. Troyansky, 2+p-SAT: relation of typical-case complexity to the nature of the phase transition, Random Struct. Algorithms 15 (1999) 414–435.
[116] B. Monien, E. Speckenmeyer, Solving satisfiability in less than 2^n steps, Discrete Appl. Math. 10 (1985) 287–295.
[117] A. Morgado, F. Heras, M.H. Liffiton, J. Planes, J. Marques-Silva, Iterative and core-guided MaxSAT solving: a survey and assessment, Constraints 18 (2013) 478–534.
[118] A. Nadel, Boosting minimal unsatisfiable core extraction, in: R. Bloem, N. Sharygina (Eds.), FMCAD, IEEE, 2010, pp. 221–229.
[119] I. Niemelä, A tableau calculus for minimal model reasoning, in: P. Miglioli, U. Moscato, D. Mundici, M. Ornaghi (Eds.), TABLEAUX, Springer, 1996, pp. 278–294.
[120] A. Nöhrer, A. Biere, A. Egyed, Managing SAT inconsistencies with HUMUS, in: U.W. Eisenecker, S. Apel, S. Gnesi (Eds.), VaMoS, ACM, 2012, pp. 83–91.
[121] C.H. Papadimitriou, Computational Complexity, Addison Wesley, 1994.
[122] D.A. Plaisted, S. Greenbaum, A structure-preserving clause form translation, J. Symb. Comput. 2 (1986) 293–304.
[123] S.D. Prestwich, CNF encodings, in: [20], 2009, pp. 75–97.
[124] A. Previti, A. Ignatiev, A. Morgado, J. Marques-Silva, Prime compilation of non-clausal formulae, in: [153], 2015, pp. 1980–1988.
[125] A. Previti, J. Marques-Silva, Partial MUS enumeration, in: M. desJardins, M.L. Littman (Eds.), AAAI, AAAI Press, 2013.
[126] P. Pritchard, Opportunistic algorithms for eliminating supersets, Acta Inform. 28 (1991) 733–754.
[127] W.V. Quine, A way to simplify truth functions, Am. Math. Mon. 62 (1955) 627–631.
[128] K. Ravi, F. Somenzi, Minimal assignments for bounded model checking, in: K. Jensen, A. Podelski (Eds.), TACAS, Springer, 2004, pp. 31–45.
[129] R. Reiter, A theory of diagnosis from first principles, Artif. Intell. 32 (1987) 57–95.
[130] R. Reiter, J. de Kleer, Foundations of assumption-based truth maintenance systems: preliminary report, in: K.D. Forbus, H.E. Shrobe (Eds.), AAAI, Morgan Kaufmann, 1987, pp. 183–189.
[131] E.D. Rosa, E. Giunchiglia, M. Maratea, Solving satisfiability problems with preferences, Constraints 15 (2010) 485–515.
[132] O. Roussel, V.M. Manquinho, Pseudo-Boolean and cardinality constraints, in: A. Biere, M. Heule, H. van Maaren, T. Walsh (Eds.), Handbook of Satisfiability, in: Front. Artif. Intell. Appl., vol. 185, IOS Press, 2009, pp. 695–733.
[133] S. Safarpour, H. Mangassarian, A.G. Veneris, M.H. Liffiton, K.A. Sakallah, Improved design debugging using maximum satisfiability, in: FMCAD, IEEE Computer Society, 2007, pp. 13–19.
[134] T. Schaub, G. Friedrich, B. O'Sullivan (Eds.), ECAI 2014 – 21st European Conference on Artificial Intelligence, 18–22 August 2014, Prague, Czech Republic – Including Prestigious Applications of Intelligent Systems, PAIS 2014, Front. Artif. Intell. Appl., vol. 263, IOS Press, 2014.
[135] N. Sharygina, H. Veith (Eds.), Computer Aided Verification, CAV, Lecture Notes in Computer Science, vol. 8044, Springer, 2013.
[136] S. Shen, Y. Qin, S. Li, A faster counterexample minimization algorithm based on refutation analysis, in: DATE, IEEE Computer Society, 2005, pp. 672–677.
[137] S. Shen, Y. Qin, S. Li, Minimizing counterexample with unit core extraction and incremental SAT, in: R. Cousot (Ed.), VMCAI, Springer, 2005, pp. 298–312.

[138] C. Sinz, A. Kaiser, W. Küchlin, Detection of inconsistencies in complex product configuration data using extended propositional SAT-checking, in: I. Russell, J.F. Kolen (Eds.), FLAIRS Conference, AAAI Press, 2001, pp. 645–649.
[139] C. Sinz, A. Kaiser, W. Küchlin, Formal methods for the validation of automotive product configuration data, Artif. Intell. Eng. Des. Anal. Manuf. 17 (2003) 75–97.
[140] J.L.N. de Siqueira, J.F. Puget, Explanation-based generalisation of failures, in: ECAI, 1988, pp. 339–344.
[141] J. Slaney, Set-theoretic duality: a fundamental feature of combinatorial optimisation, in: [134], 2014, pp. 843–848.
[142] J.K. Slaney, T. Walsh, Backbones in optimization and approximation, in: B. Nebel (Ed.), IJCAI, Morgan Kaufmann, 2001, pp. 254–259.
[143] T. Soh, K. Inoue, Identifying necessary reactions in metabolic pathways by minimal model generation, in: H. Coelho, R. Studer, M. Wooldridge (Eds.), ECAI, IOS Press, 2010, pp. 277–282.
[144] J. Straach, K. Truemper, Learning to ask relevant questions, Artif. Intell. 111 (1999) 301–327.
[145] S.T. To, T.C. Son, E. Pontelli, Conjunctive representations in contingent planning: prime implicates versus minimal CNF formula, in: W. Burgard, D. Roth (Eds.), AAAI, AAAI Press, 2011.
[146] G.S. Tseitin, On the complexity of derivation in propositional calculus, in: Studies in Constructive Mathematics and Mathematical Logic, Part II, 1968, pp. 115–125.
[147] C. Umans, T. Villa, A.L. Sangiovanni-Vincentelli, Complexity of two-level logic minimization, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 25 (2006) 1230–1246.
[148] A. Van Gelder, Autarky pruning in propositional model elimination reduces failure redundancy, J. Autom. Reason. 23 (1999) 137–193.
[149] A. Van Gelder, Complexity analysis of propositional resolution with autarky pruning, Discrete Appl. Math. 96–97 (1999) 195–221.
[150] S. Vesic, Identifying the class of maxi-consistent operators in argumentation, J. Artif. Intell. Res. 47 (2013) 71–93.
[151] J.P. Wallner, G. Weissenbacher, S. Woltran, Advanced SAT techniques for abstract argumentation, in: J. Leite, T.C. Son, P. Torroni, L. van der Torre, S. Woltran (Eds.), CLIMA, Springer, 2013, pp. 138–154.
[152] S. Wieringa, Understanding, improving and parallelizing MUS finding using model rotation, in: M. Milano (Ed.), CP, Springer, 2012, pp. 672–687.
[153] Q. Yang, M. Wooldridge (Eds.), Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25–31, 2015, AAAI Press, 2015.
[154] A. Zeller, Yesterday, my program worked. Today, it does not. Why?, in: O. Nierstrasz, M. Lemoine (Eds.), ESEC/SIGSOFT FSE, Springer, 1999, pp. 253–267.
[155] J. Zhang, S. Shen, J. Zhang, W. Xu, S. Li, Extracting minimal unsatisfiable subformulas in satisfiability modulo theories, Comput. Sci. Inf. Syst. 8 (2011) 693–710.
[156] C.S. Zhu, G. Weissenbacher, S. Malik, Post-silicon fault localisation using maximum satisfiability and backbones, in: P. Bjesse, A. Slobodová (Eds.), FMCAD, FMCAD Inc., 2011, pp. 63–66.
[157] C.S. Zhu, G. Weissenbacher, D. Sethi, S. Malik, SAT-based techniques for determining backbones for post-silicon fault localisation, in: Z. Zilic, S.K. Shukla (Eds.), HLDVT, IEEE, 2011, pp. 84–91.
[158] C. Zielke, M. Kaufmann, A new approach to partial MUS enumeration, in: [58], 2015, pp. 387–404.