An Algorithm for Enumerating Maximal Models of Horn Theories with an Application to Modal Logics*

Luca Aceto[1], Dario Della Monica[1], Anna Ingólfsdóttir[1], Angelo Montanari[2], and Guido Sciavicco[3]

[1] ICE-TCS, School of Computer Science
Reykjavik University, Reykjavik, Iceland – {luca,dariodm,annai}@ru.is
[2] Department of Mathematics and Computer Science
University of Udine, Udine, Italy – angelo.montanari@uniud.it
[3] Department of Information, Engineering and Communications
University of Murcia, Murcia, Spain – guido@um.es

Abstract. The fragment of propositional logic known as Horn theories plays a central role in automated reasoning. The problem of enumerating the maximal models of a Horn theory (MaxMod) has been proved to be computationally hard, unless P = NP. To the best of our knowledge, the only algorithm available for it is the one based on a brute-force approach. In this paper, we provide an algorithm for the problem of enumerating the maximal subsets of facts that do not entail a distinguished atomic proposition in a positive Horn theory (MaxNoEntail). We show that MaxMod is polynomially reducible to MaxNoEntail (and vice versa), making it possible to solve also the former problem using the proposed algorithm. Addressing MaxMod via MaxNoEntail opens, inter alia, the possibility of benefiting from the monotonicity of the notion of entailment. (The notion of model does not enjoy such a property.) We also discuss an application of MaxNoEntail to expressiveness issues for modal logics, which reveals the effectiveness of the proposed algorithm.

1 Introduction

Propositional logic is the most basic tool in computer science and artificial intelligence. Despite its limited expressive power, it allows one to formalize several interesting scenarios. In particular, the fragment of propositional logic known as Horn theories [8] plays a central role in the search for efficient reasoning methods thanks to its good computational properties: the entailment problem can be

* The authors acknowledge the support from the Spanish fellowship program *'Ramon y Cajal' RYC-2011-07821* and the Spanish MEC project *TIN2009-14372-C03-01* (G. Sciavicco), the project *Processes and Modal Logics* (project nr. 100048021) of the Icelandic Research Fund (L. Aceto, D. Della Monica, and A. Ingólfsdóttir), the project *Decidability and Expressiveness for Interval Temporal Logics* (project nr. 130802-051) of the Icelandic Research Fund (D. Della Monica), and the Italian GNCS project *Extended Game Logics* (A. Montanari).

solved in linear time [7,14], while it is NP-complete for full propositional logic. A Horn theory is a conjunction of clauses (that is, disjunctions of literals) such that every clause has, at most, one positive literal.

Horn theories can be applied to a number of different fields, such as planning [11], case based reasoning [15], or diagnosis [4]. A common problem is that of enumerating the models of a given theory with a particular property, e.g., maximality or minimality. As an example, the concepts of propositional circumscription and minimal/maximal diagnosis are related to this problem [5,6]. A model of a Horn theory is a truth assignment for all its atomic propositions that satisfies the theory. A model is maximal if extending its set of true propositions has the effect of losing the property of being a model. The problem of enumerating the maximal models of a given Horn theory, called here MaxMod, has been studied in [12]. Since the problem has, in general, an output whose dimension (number of solutions returned) is exponential in the size of the input, one can hope, at best, to have an output-polynomial algorithm, that is, an algorithm whose complexity is polynomial in the size of both input and output. (A survey on the relationship between the output complexity hierarchy and the classical complexity hierarchy can be found in [13,16].) In [12], it has been proved that, unless P=NP, no output-polynomial algorithm can be devised for MaxMod. This discouraged further investigation in the search for efficient algorithms for MaxMod. As a consequence, to the best of our knowledge, the only algorithm available for it is the one based on a brute-force approach. It explores the space of truth assignments over the set of atomic propositions searching for maximal models. The trivial way to do so is in two steps: first, by identifying those assignments that are models, and then by checking them for maximality. Since the number of assignments is the size of the powerset of the set of propositions, the algorithm runs in exponential time.

In this paper, we establish a connection between MaxMod and the problem of enumerating all maximal subsets of atomic propositions (facts) that do not entail a distinguished proposition in a given *positive* Horn theory (a theory where all clauses contain exactly one positive literal). The outcome of the latter problem, called here MaxNoEntail, can be intuitively interpreted as follows: *all maximal sets of causes that do not have atomic proposition X as a consequence*. We show that MaxMod and MaxNoEntail are polynomially equivalent; thus, every algorithm for MaxNoEntail is also an algorithm for MaxMod. It is worth noticing that the notion of entailment is monotone: if a set of facts entails a proposition, also each of its extensions does. Consequently, in order to check the maximality of a set F of facts that do not entail a given proposition X in a positive Horn theory, it is enough to check that every extension obtained by adding a *single* new proposition to F *does* entail X. On the other hand, the notion of model (and thus the notion of *non-model*) does not enjoy a similar property and thus, in order to verify the maximality of a model M of a Horn theory, it is necessary to verify that all the valuations *extending* M (i.e., the valuations for which the set of true propositions is an extension of the set of true propositions of M) are not models of the theory. Thanks to the mono-

tonicity of entailment, the brute-force algorithm for MaxNoEntail performs better than the brute-force approach for MaxMod. Thus, reducing MaxMod to MaxNoEntail immediately gives us a faster, yet trivial, solution to MaxMod. Furthermore, we present an alternative algorithm for MaxNoEntail that performs better than the brute-force approach, as it minimizes the number of candidate solutions that are tested before producing the next solution.

Another benefit resulting from approaching MaxMod via MaxNoEntail is that the latter problem is closely related to expressiveness issues for modal logics [3]. Indeed, such a relation between Horn theories and modal logics motivated this study in the first place [1,9]. A major issue in modal logic is that of finding out which modalities can be expressed in terms of others, in order to classify all expressively different sub-logics with respect to, e.g., expressive power or complexity of the satisfiability problem. A common approach to this problem consists of two steps: first, identifying as many inter-definabilities as possible, and then trying to prove completeness of such a set of inter-definabilities. The second step has two possible outcomes: either one is able to prove completeness, or the failure in proving it might suggest new inter-definabilities, giving rise to a new, extended set of inter-definabilities to be checked for completeness. In any case, the second step requires the identification of all maximal subsets of modalities that, within the current set of known inter-definabilities, do *not* express a specific modality. Since a set of inter-definabilities between modalities can be thought of as a positive Horn theory (where atomic propositions play the role of the modalities), identifying such maximal subsets of modalities amounts to solving MaxNoEntail. We provide empirical evidence that the proposed algorithm for MaxNoEntail is particularly efficient when applied to the study of the expressive power of modal logics, as described above, allowing us to solve instances that were intractable with the brute-force approach.

The paper is organized as follows. In Section 2, we give the preliminaries. In Section 3, we prove that MaxMod and MaxNoEntail are polynomially equivalent. We also present there the brute-force algorithm for MaxNoEntail, that gives us a more efficient solution for MaxMod. In Section 4, we present an alternative algorithm for MaxNoEntail and we prove its correctness. In Section 5, we give evidence of the effectiveness of the proposed method when applied to expressiveness issues for modal logics. Finally, in Section 6, we give an assessment of the work and outline future research directions.

2 Preliminaries

Throughout the paper, $\mathcal{P}$ denotes a finite, non-empty set of atomic propositions. A *Boolean expression* over $\mathcal{P}$ is a formula built using propositions from $\mathcal{P}$ and the classic Boolean operators of negation, conjunction, and disjunction. Every Boolean expression can be transformed into an equivalent formula in *conjunctive normal form* (CNF), where the outermost operator is the conjunction and each conjunct is a disjunction of *literals*, that is, atomic propositions (*positive literals*) or their negation (*negative literals*). A *Horn theory* (or *Horn expression*) over $\mathcal{P}$ is

a Boolean expression over $\mathcal{P}$ in CNF whose conjuncts have at most one positive literal. Conjuncts of a Horn theory are referred to as *clauses*. It is common practice to think of a Horn theory $\mathcal{K}$ as the set $\{\delta_1, \ldots, \delta_k\}$ of its clauses. The atomic propositions occurring negated in a clause are called *antecedents* of the clause; the positive literal, if any, is called *consequent* of the clause. A clause $\delta_i = \neg A^i_1 \vee \ldots \vee \neg A^i_{m_i} \vee A^i$ of a Horn theory can be seen as the implication of the consequent by the antecedents, written as $A^i_1, \ldots, A^i_{m_i} \Rightarrow A^i$. A clause with exactly one literal is a *fact*. A clause $\neg A^i_1 \vee \ldots \vee \neg A^i_{m_1}$ with no positive literal can be seen as $A^i_1, \ldots, A^i_{m_i} \Rightarrow \bot$. Thus, it is useful to think of $\bot$ as a distinguished atomic proposition in $\mathcal{P}$, whose truth value is 0 in each truth assignment (see below for a formal definition of the notion of assignment). A theory in which every clause contains exactly one positive literal is said to be *positive*. Given a clause δ, we denote by ant_δ its set of antecedents, and by $cons_\delta$ the singleton containing the consequent. Finally, by $HT_\mathcal{P}$ (resp., $PHT_\mathcal{P}$), we denote the set of all (resp., positive) Horn theories over the set of atomic propositions $\mathcal{P}$.

An *assignment* M over $\mathcal{P}$ is defined as a function $M : \mathcal{P} \to \{0, 1\}$, assigning a truth value to every proposition in $\mathcal{P}$. An assignment M over $\mathcal{P}$ is a *model* of a Horn theory $\mathcal{K} \in HT_\mathcal{P}$, denoted by $M \models \mathcal{K}$, if and only if it satisfies all the clauses of $\mathcal{K}$. A Horn theory is *satisfiable* if and only if there exists a model for it. Moreover, we say that $\mathcal{K}$ *entails* a literal l, denoted $\models_\mathcal{K} l$, if and only if $\mathcal{K} \cup \{\neg l\}$ is not satisfiable. Here, we are mainly interested in entailment of *positive* literals. Given a Horn theory $\mathcal{K} \in HT_\mathcal{P}$, a subset of $\mathcal{P}$ is also referred to as a *fragment* (of $\mathcal{P}$). Thus, a fragment is a set of positive literals. Given a fragment F of $\mathcal{P}$, a positive literal $X \in \mathcal{P}$, and a Horn theory $\mathcal{K} \in HT_\mathcal{P}$, we say that F *entails* X *in* $\mathcal{K}$, denoted by $F \models_\mathcal{K} X$, if and only if $\mathcal{K} \cup F \cup \{\neg X\}$ is unsatisfiable, that is, every model M of $\mathcal{K} \cup F$ is such that $M(X) = 1$. Given an assignment M over $\mathcal{P}$, we define the fragment *induced* by M, denoted by $\eta(M)$, as the one containing exactly the propositions that are true in M. On the other hand, given a fragment F of $\mathcal{P}$, the assignment *induced* by F, denoted by $\mu(F)$, is obtained by setting to 1 the propositions in F, and to 0 the ones in $\mathcal{P} \setminus F$. It obviously holds that $F = \eta(\mu(F))$ and $M = \mu(\eta(M))$, for each fragment F of $\mathcal{P}$ and for each assignment M over $\mathcal{P}$. The notion of entailment can now be extended from fragments to assignments: M *entails* X in $\mathcal{K}$, denoted by $M \models_\mathcal{K} X$, if and only if $\eta(M) \models_\mathcal{K} X$. Similarly, the order over fragments induced by the set inclusion operation $\subset$ can be extended to assignments as follows: $M \prec M'$ if and only if $\eta(M) \subset \eta(M')$. Notice also that entailment is monotonic: if $F \models_\mathcal{K} X$ (resp., $M \models_\mathcal{K} X$) holds for some fragment F (resp., model M), then $F' \models_\mathcal{K} X$ (resp., $M' \models_\mathcal{K} X$) holds for every F' such that $F \subset F'$ (resp., M' such that $M \prec M'$).

Given a Horn theory $\mathcal{K}$, a model M of $\mathcal{K}$ is *maximal* if and only if $M' \not\models \mathcal{K}$ for every assignment M' such that $M \prec M'$. A fragment F is *X-incomplete in* $\mathcal{K}$ if and only if $F \not\models_\mathcal{K} X$, and it is *maximally X-incomplete in* $\mathcal{K}$ if and only if it is *X-incomplete in* $\mathcal{K}$ and $F' \models_\mathcal{K} X$ for every fragment F' such that $F \subset F'$. We will sometimes omit the specification of the Horn theory if it is clear from the context. Clearly, the monotonicity of entailment implies the monotonicity of X-incompleteness (if F is X-incomplete, then each of its subsets is X-incomplete,

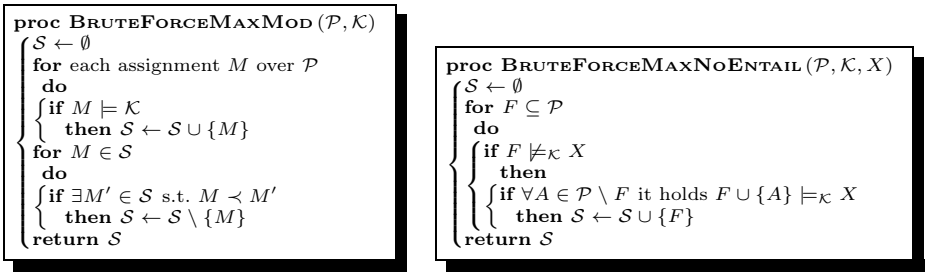

Fig. 1. The brute-force algorithm for MaxMod (left-hand side), and the one, more efficient, for MaxNoEntail (right-hand side).

as well). Therefore, the notion of maximal incompleteness can be rephrased in the following equivalent way: F is maximally X-incomplete if and only if it is X-incomplete and $F \cup \{A\} \models_{\mathcal{K}} X$ for each $A \in \mathcal{P} \setminus F$. On the contrary, the notion of model of a generic theory does not enjoy such a property. As an example, consider the theory $\mathcal{K}$, featuring the only clause $A, B \Rightarrow C$: the assignment M, which sets all the propositions to 0, is a model of $\mathcal{K}$; the assignment M', which extends the set of true propositions of M by setting A and B to 1, is not a model of $\mathcal{K}$; the assignment M'', which in turn extends the set of true propositions of M' by setting also C to 1, is another model of $\mathcal{K}$.

We are now ready to formally define the enumeration problems MaxMod and MaxNoEntail, that are the aim of this study.

Definition 1. *Given a set of atomic propositions $\mathcal{P}$ and a Horn theory $\mathcal{K} \in HT_{\mathcal{P}}$, the problem* MaxMod *is defined as the problem of enumerating all and only the assignments over $\mathcal{P}$ that are maximal models of $\mathcal{K}$. Similarly, given a set of atomic propositions $\mathcal{P}$, a positive Horn theory $\mathcal{K} \in PHT_{\mathcal{P}}$, and a distinguished atomic proposition $X \in \mathcal{P}$, the problem* MaxNoEntail *is defined as the problem of enumerating all and only the fragments F of $\mathcal{P}$ that are maximally X-incomplete in $\mathcal{K}$.*

For the sake of completeness, before concluding the section we provide, in Fig. 1, left-hand side, the pseudo-code of a trivial, brute-force algorithm for MaxMod. It is clear that the algorithm described there is highly inefficient, and obviously not output-polynomial (in [12] it is proven that, unless P=NP, no output-polynomial algorithm exists for this problem): even if the set of solutions is small, or even empty, the algorithm requires an exponential number of steps. Moreover, the algorithm performs two iterations: the one on the space of the valuations over $\mathcal{P}$, whose size is exponential in the one of the input, and the other on the space of the models of the Horn theory $\mathcal{K}$, whose size is possibly exponential in the one of the input, as well. In what follows, we first present a brute-force algorithm for MaxNoEntail (see Fig. 1, right-hand side) that, thanks to the monotonicity of entailment, avoids the second iteration step, thus having better performance than the one for MaxMod. Then, we propose a more efficient

solution for MaxNoEntail. Since, as we will show, MaxMod is polynomially reducible to MaxNoEntail, the proposed algorithms for MaxNoEntail apply to MaxMod, too.

3 Solving MaxMod through MaxNoEntail

In this section, we provide a polynomial reduction from MaxMod to MaxNoEntail, and the other way around. This allows us to employ the brute-force algorithm for MaxNoEntail, depicted in Fig. 1 (right-hand side), to solve MaxMod, thus obtaining a more efficient, yet trivial, solution for it that benefits from the monotonicity of entailment. A MaxMod *instance* is a pair $\langle \mathcal{P}, \mathcal{K} \rangle$, where $\mathcal{P}$ is a set of propositions and $\mathcal{K} \in HT_{\mathcal{P}}$. A MaxNoEntail *instance* is a triple $\langle \mathcal{P}, \mathcal{K}, X \rangle$, where $\mathcal{P}$ is a set of propositions, $\mathcal{K} \in PHT_{\mathcal{P}}$, and $X \in \mathcal{P}$. In what follows, we define the functions τ and γ that are used to transform MaxMod instances into MaxNoEntail ones, and vice versa.

Definition 2. *$\tau : HT_{\mathcal{P}} \rightarrow PHT_{\mathcal{P} \cup \{X\}}$, where X is a distinguished atomic proposition not belonging to $\mathcal{P}$, is defined as follows: for each Horn theory $\mathcal{K} \in HT_{\mathcal{P}}$, $\tau(\mathcal{K})$ is the smallest theory such that: (i) for each clause $\delta \in \mathcal{K}$ that contains one positive literal, δ belongs to $\tau(\mathcal{K})$, and (ii) for each clause $\delta \in \mathcal{K}$ of the type $ant_\delta \Rightarrow \bot$ (i.e., δ does not contain positive literals), the clause $ant_\delta \Rightarrow X$ belongs to $\tau(\mathcal{K})$. $\gamma : PHT_{\mathcal{P}} \times \mathcal{P} \rightarrow HT_{\mathcal{P}}$ is defined as follows: for each positive Horn theory $\mathcal{K} \in PHT_{\mathcal{P}}$ and proposition $X \in \mathcal{P}$, $\gamma(\mathcal{K}, X) = \mathcal{K} \cup \{\neg X\}$.*

Our goal is to show that, for every MaxMod instance $\langle \mathcal{P}, \mathcal{K} \rangle$, with $X \notin \mathcal{P}$, the set of solutions of MaxMod on $\langle \mathcal{P}, \mathcal{K} \rangle$ coincides with the set of solutions of MaxNoEntail on $\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle$, and that, for every MaxNoEntail instance $\langle \mathcal{P}, \mathcal{K}, X \rangle$, the set of solutions of MaxNoEntail on $\langle \mathcal{P}, \mathcal{K}, X \rangle$ coincides with the set of solutions of MaxMod on $\langle \mathcal{P}, \gamma(\mathcal{K}, X) \rangle$. Let us give, first, a technical lemma (whose proof is given in Appendix A).

Lemma 1. *Let $\mathcal{K} \in HT_{\mathcal{P}}$ and $A \in \mathcal{P}$. The following results hold.*

(a) Let F be a fragment of $\mathcal{P}$ that is maximally X-incomplete in $\mathcal{K}$. Then, $A \in F$ if and only if $F \models_{\mathcal{K}} A$.
(b) Let M be a model of $\mathcal{K}$. Then, $M(A) = 1$ if and only if $M \models_{\mathcal{K}} A$.

Let us denote by $\mathcal{M}_{\langle \mathcal{P}, \mathcal{K} \rangle}$ the set of solutions for MaxMod on the generic instance $\langle \mathcal{P}, \mathcal{K} \rangle$ and by $\mathcal{I}_{\langle \mathcal{P}, \mathcal{K}, X \rangle}$ the set of solutions for MaxNoEntail on the generic instance $\langle \mathcal{P}, \mathcal{K}, X \rangle$. In the following two lemmas, we prove that MaxMod is reducible to MaxNoEntail (Lemma 2) and vice versa (Lemma 3).

Lemma 2. *Let $\langle \mathcal{P}, \mathcal{K} \rangle$ be a generic instance of* MaxMod*, with $X \notin \mathcal{P}$. Then, $\mathcal{M}_{\langle \mathcal{P}, \mathcal{K} \rangle} = \{\mu(F) \mid F \in \mathcal{I}_{\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle}\}$.*

Proof. We proceed in two steps: first, we show that $\mu(F) \in \mathcal{M}_{\langle \mathcal{P}, \mathcal{K} \rangle}$, for each $F \in \mathcal{I}_{\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle}$; then, we prove that, for each model $M \in \mathcal{M}_{\langle \mathcal{P}, \mathcal{K} \rangle}$, there exists a fragment $F \in \mathcal{I}_{\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle}$ such that $\mu(F) = M$.

To prove the former claim, let us assume $F \in \mathcal{I}_{\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle}$, which means that F is maximally X-incomplete in $\tau(\mathcal{K})$. We want to show that $\mu(F)$ belongs to $\mathcal{M}_{\langle \mathcal{P}, \mathcal{K} \rangle}$, that is, $\mu(F)$ is a maximal model for $\mathcal{K}$.

- To prove that $\mu(F)$ is a model of $\mathcal{K}$, i.e., $\mu(F) \models \mathcal{K}$, let δ be a clause of $\mathcal{K}$. We shall argue show that $\mu(F)$ satisfies δ. We distinguish two cases.
  - δ is of the form $ant_\delta \Rightarrow A$, for some $A \in \mathcal{P}$. If $\mu(F)$ does not satisfy ant_δ, then we are done. Assume that $\mu(F)$ does satisfy ant_δ. We shall show that $\mu(F)(A) = 1$. Since $\mu(F)$ satisfies ant_δ, we have that $ant_\delta \subseteq F$. This means that $\{\delta\} \cup F \cup \{\neg A\}$ is unsatisfiable and thus $F \models_{\tau(\mathcal{K})} A$ holds, because δ is also a clause of $\tau(\mathcal{K})$, by construction. By Lemma 1(a), $A \in F$ and therefore $\mu(F)(A) = 1$, as claimed.
  - δ is of the form $ant_\delta \Rightarrow \bot$. We claim that $\mu(F)$ does not satisfy ant_δ. To see this, let us assume, towards a contradiction, that $\mu(F)$ satisfies ant_δ. Then, $ant_\delta \subseteq F$. By construction of $\tau(\mathcal{K})$, the clause $ant_\delta \Rightarrow X$ belongs to $\tau(\mathcal{K})$. Now, we have that $\{\delta\} \cup F \cup \{\neg X\}$ is unsatisfiable and thus $F \models_{\tau(\mathcal{K})} X$ holds, contradicting the X-incompleteness of F.

  Since $\mu(F)$ satisfies each clause of $\mathcal{K}$, we have that $\mu(F) \models \mathcal{K}$ holds.
- To prove the maximality of $\mu(F)$, let us assume, towards a contradiction, that there exists a model M of $\mathcal{K}$ such that $\mu(F) \prec M$. By the definition of $\eta(\cdot)$, this implies $F \subset \eta(M)$. We claim that $\eta(M)$ is X-incomplete in $\tau(\mathcal{K})$, thus obtaining a contradiction with the fact that F is maximally X-incomplete. Indeed, since M is a model of $\mathcal{K}$, it does not satisfy any of the sets ant_δ, where $\delta \in \mathcal{K}$ is of the form $ant_\delta \Rightarrow \bot$. Thus, $\eta(M) \not\supseteq ant_\delta$, for every $\delta \in \tau(\mathcal{K})$ of the form $ant_\delta \Rightarrow X$, which yields $\eta(M) \not\models_{\mathcal{K}} X$. This, in turn, means that $\eta(M)$ is X-incomplete in $\tau(\mathcal{K})$, which contradicts the maximality of F.

To complete the proof, let us consider a model $M \in \mathcal{M}_{\langle \mathcal{P}, \mathcal{K} \rangle}$, that is, M is a maximal model of $\mathcal{K}$. Our aim is to show that there exists a fragment $F \in \mathcal{I}_{\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle}$ such that $\mu(F) = M$. We claim that $\eta(M) \in \mathcal{I}_{\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle}$. Since $\mu(\eta(M)) = M$, the thesis follows from this claim. First, we prove that $\eta(M)$ is X-incomplete in $\tau(\mathcal{K})$, i.e., $\eta(M) \not\models_{\tau(\mathcal{K})} X$. To this end, let M' be the valuation over $\mathcal{P} \cup \{X\}$ obtained from M as follows: $M'(Y) = M(Y)$ for each $Y \in \mathcal{P}$ and $M'(X) = 0$. It is easy to see that M' is a model for $\tau(\mathcal{K}) \cup \eta(M) \cup \{\neg X\}$. Thus, $\tau(\mathcal{K}) \cup \eta(M) \cup \{\neg X\}$ is satisfiable, which implies $\eta(M) \not\models_{\tau(\mathcal{K})} X$. Now, in order to prove that $\eta(M)$ is maximally X-incomplete, we have to show that $\eta(M) \cup \{A\} \models_{\tau(\mathcal{K})} X$, for each $A \in (\mathcal{P} \cup \{X\}) \setminus \eta(M)$. If $A = X$, the thesis trivially follows from the definition of entailment. Otherwise, let us suppose, towards a contradiction, that $\eta(M) \cup \{A\} \not\models_{\tau(\mathcal{K})} X$, for some $A \in (\mathcal{P} \cup \{X\}) \setminus \eta(M)$, with $A \neq X$. This means that $\tau(\mathcal{K}) \cup \eta(M) \cup \{A\} \cup \{\neg X\}$ is satisfiable. Let M' be a model for it. Since M' is a model of $\tau(\mathcal{K})$ and $M'(X) = 0$, it is also a model of $\mathcal{K}$ (by construction of $\tau(\mathcal{K})$, X syntactically replaces the symbol $\bot$). Moreover, it is easy to convince oneself that $\eta(M') \supseteq \eta(M) \cup \{A\}$. Thus M' is a model of $\mathcal{K}$ such that $M \prec M'$, contradicting the maximality of M. Hence $\eta(M)$ is maximally X-incomplete in $\tau(\mathcal{K})$, and the thesis follows. □

Lemma 3. *Let* $\langle \mathcal{P}, \mathcal{K}, X \rangle$ *be a generic instance of* MaxNoEntail. *Then,* $\mathcal{I}_{\langle \mathcal{P},\mathcal{K},X \rangle} = \{\eta(M) \mid M \in \mathcal{M}_{\langle \mathcal{P},\gamma(\mathcal{K},X)\rangle}\}$.

Proof. We prove the statement in two steps: first, we show that $\eta(M) \in \mathcal{I}_{\langle \mathcal{P},\mathcal{K},X \rangle}$, for each $M \in \mathcal{M}_{\langle \mathcal{P},\gamma(\mathcal{K},X)\rangle}$; then, we show that, for each fragment $F \in \mathcal{I}_{\langle \mathcal{P},\mathcal{K},X \rangle}$, there exists a model $M \in \mathcal{M}_{\langle \mathcal{P},\gamma(\mathcal{K},X)\rangle}$ such that $\eta(M) = F$.

To prove the former claim, let us assume $M \in \mathcal{M}_{\langle \mathcal{P},\gamma(\mathcal{K},X)\rangle}$, which means that M is a maximal model of $\gamma(\mathcal{K}, X)$. As a preliminary step, we observe that, by construction of $\gamma(\mathcal{K}, X)$, every model of $\gamma(\mathcal{K}, X)$ is also a model of $\mathcal{K}$. We want to show that $\eta(M)$ belongs to $\mathcal{I}_{\langle \mathcal{P},\mathcal{K},X \rangle}$, that is, $\eta(M)$ is maximally X-incomplete in $\mathcal{K}$. First, we show that $\eta(M)$ is X-incomplete in $\mathcal{K}$, and then that it is maximally X-incomplete in $\mathcal{K}$. To show the X-incompleteness of $\eta(M)$, suppose, towards a contradiction, that $\eta(M) \models_{\mathcal{K}} X$. This means that $M \models_{\mathcal{K}} X$ and, by Lemma 1(b) and by the fact that M is also a model of $\mathcal{K}$, it follows that $M(X) = 1$, which implies that M is not a model of $\gamma(\mathcal{K}, X)$. This contradicts the assumption that $M \in \mathcal{M}_{\langle \mathcal{P},\gamma(\mathcal{K},X)\rangle}$. So, we have that $\eta(M)$ is X-incomplete in $\mathcal{K}$. Now, suppose, towards a contradiction, that $\eta(M)$ is not maximally X-incomplete. Thus, $\eta(M) \cup \{A\} \not\models_{\mathcal{K}} X$ holds, for some $A \in \mathcal{P} \setminus \eta(M)$. This means that $\mathcal{K} \cup \eta(M) \cup \{A\} \cup \{\neg X\}$ is satisfiable. Let M' be a model for it. Since M' satisfies $\mathcal{K}$ and $\{\neg X\}$, it is also a model of $\gamma(\mathcal{K}, X)$. Moreover, it is easy to see that $\eta(M) \subset \eta(M')$. Thus, M' is a model of $\gamma(\mathcal{K}, X)$ such that $M \prec M'$, which contradicts the maximality of M.

To complete the proof, let us consider a fragment $F \in \mathcal{I}_{\langle \mathcal{P},\mathcal{K},X \rangle}$, that is, F is maximally X-incomplete in $\mathcal{K}$. Our goal is to show that there exists a model $M \in \mathcal{M}_{\langle \mathcal{P},\gamma(\mathcal{K},X)\rangle}$ such that $\eta(M) = F$. We claim that $\mu(F) \in \mathcal{M}_{\langle \mathcal{P},\gamma(\mathcal{K},X)\rangle}$. Since $\eta(\mu(F)) = F$, the thesis follows from this claim. First, we show that $\mu(F)$ is a model of $\gamma(\mathcal{K}, X)$. By construction, $\gamma(\mathcal{K}, X) = \mathcal{K} \cup \{\neg X\}$. By Lemma 1(a) and by the assumption that F is maximally X-incomplete, it follows $X \notin F$, which means that $\mu(F)(X) = 0$. Thus, $\mu(F)$ satisfies $\{\neg X\}$. Now, let us show that $\mu(F)$ also satisfies $\mathcal{K}$. Let δ be a generic clause in $\mathcal{K}$. It is of the form $A_1, \ldots, A_m \Rightarrow A$. We distinguish two cases. If $A_i \notin F$ for some $i \in \{1, \ldots, m\}$, then $\mu(F)(A_i) = 0$, which means that δ is satisfied by $\mu(F)$. Otherwise, $\{A_1, \ldots, A_m\} \subseteq F$, which means that $F \models_{\mathcal{K}} A$. Therefore, by Lemma 1(a), $A \in F$, which implies that $\mu(F)(A) = 1$. So, $\mu(F) \models \delta$ and, since δ was chosen arbitrarily, $\mu(F)$ is a model of $\mathcal{K}$. Since we showed that it is also a model of $\{\neg X\}$, we have that $\mu(F)$ is a model of $\gamma(\mathcal{K}, X)$. To prove the maximality of $\mu(F)$, let us suppose, towards a contradiction, that there exists a model M' of $\gamma(\mathcal{K}, X)$, such that $\mu(F) \prec M'$, which means $F \subset \eta(M')$. Since M' is a model of $\gamma(\mathcal{X}, X)$, it is both a model of $\mathcal{K}$ and $\{\neg X\}$. In particular, the latter implies $M'(X) = 0$. By Lemma 1(b), $M' \not\models_{\mathcal{K}} X$ holds, which means $\eta(M') \not\models_{\mathcal{K}} X$. Thus, $\eta(M')$ is a fragment that is X-incomplete in $\mathcal{K}$ such that $F \subset \eta(M')$. This contradicts the assumption that F is maximally X-incomplete. Hence, $\mu(F)$ is a maximal model of $\gamma(\mathcal{K}, X)$. □

The following theorem follows from Lemma 2, Lemma 3, and Definition 2.

Theorem 1. MaxMod *and* MaxNoEntail *are polynomially equivalent.*

Thanks to the above reduction, it is possible to use the brute-force algorithm for MaxNoEntail, depicted in Fig. 1, right-hand side, to solve MaxMod. While it is still based on a brute-force approach, such an algorithm turns out to be much more effective than the one described in Fig. 1, left-hand side. Indeed, in searching for fragments that are maximally X-incomplete in the given theory, one can easily verify the maximality of a candidate (i.e., an X-incomplete fragment) by checking if adding exactly one element to it preserves its incompleteness. This allows us to avoid a second pass on the set of potential results.

4 An algorithm for MaxNoEntail

In this section we present an alternative algorithm for MaxNoEntail, called *AlgMaxNoEn* (see Fig. 2). We prove that our algorithm is correct and, in the next section, we give experimental evidence of its effectiveness when applied to expressiveness issues for modal logics (see the discussion in Section 1).

We begin by giving some definitions that will be useful in what follows. Since only positive Horn theories occur in MaxNoEntail instances, throughout the section we assume that all Horn theories are positive, unless otherwise specified.

Definition 3. *Let $\mathcal{K} \in PHT_{\mathcal{P}}$ be a Horn theory, δ be a clause, and F be a fragment of $\mathcal{P}$, with $A \in F$. We say that: (i) A* deactivates *δ if A belongs to ant_δ; (ii) A is (F, δ)-*useful *if A deactivates δ and no other proposition in F does; (iii) A is $(F, \mathcal{K})$-*useful *if A is (F, δ')-useful for some $\delta' \in \mathcal{K}$; (iv) F is $\mathcal{K}$-*useful *if X is $(F, \mathcal{K})$-useful for every $X \in F$.*

Notice that, for a given clause δ and fragment F, there can be at most one proposition in F that is (F, δ)-useful. More precisely, such a proposition exists if and only if $|F \cap ant_\delta| = 1$. In what follows, we will simply say that F is useful (in place of $\mathcal{K}$-useful) when the theory is clear from the context. The important property relating the notions of maximal X-incompleteness and usefulness is stated by the following lemma.

Lemma 4. *If a fragment F of $\mathcal{P}$ is maximally X-incomplete in $\mathcal{K}$, then its complement $\mathcal{P} \setminus F$ is useful.*

Notice that the the converse does not necessarily hold. Indeed, if F is X-incomplete and its complement is useful, then F is not necessarily maximally X-incomplete. As an example, consider the theory $\mathcal{K} = \{A, B \Rightarrow X, C \Rightarrow A\}$. The fragment A is X-incomplete and its complement BC is useful, but A is not maximally X-incomplete, as AC is X-incomplete, as well. Moreover, observe that, if a fragment F is not useful, then any fragment F' such that $F \subset F'$ is not useful, either. This follows from the fact that if a proposition $A \in F$ is not $(F, \mathcal{K})$-useful, then it is not $(F', \mathcal{K})$-useful for any F' such that $F \subset F'$.

Definition 4. *Given a Horn theory $\mathcal{K} = \{\delta_1, \ldots, \delta_k\}$ and a fragment F, the* utility vector *of F in $\mathcal{K}$, usually denoted by $\boldsymbol{u}$, is a vector of size k such that, for each index i, $\boldsymbol{u}[i]$ is equal to $\{A\}$ if A is (F, δ_i)-useful, and it is equal to* `null` *if $|F \cap ant_{\delta_i}| \neq 1$.*

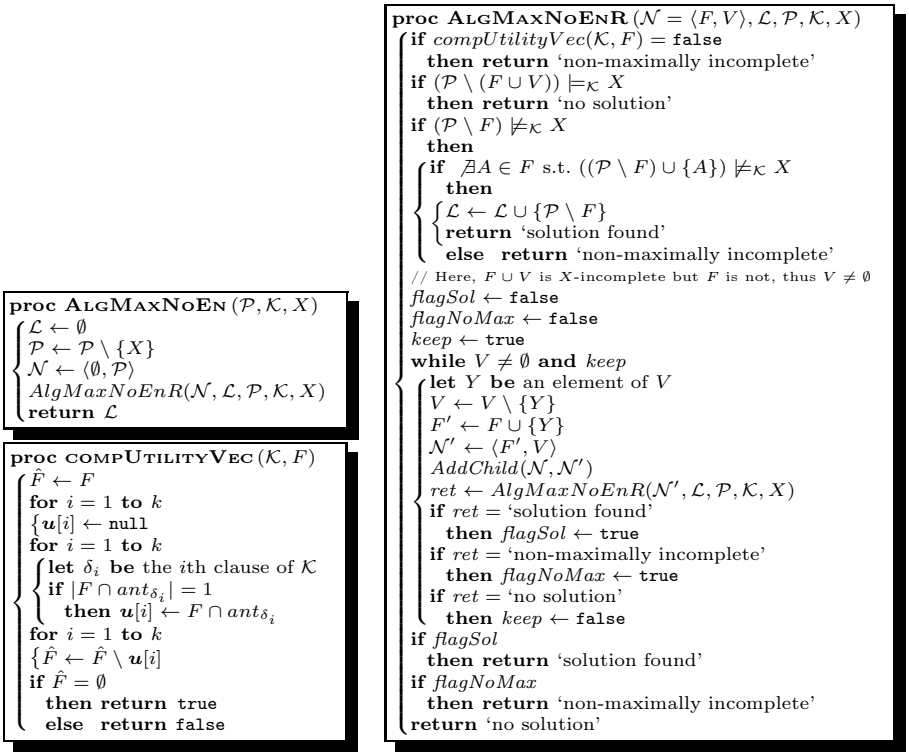

Fig. 2. Pseudo-code for the algorithms *AlgMaxNoEn* (left-hand side, top), *compUtilityVec* (left-hand side, bottom), and *AlgMaxNoEnR* (right-hand side).

Intuitively, the utility vector is the tool used to detect that a fragment is not useful: F is useful if and only if all the propositions in F occur in $\boldsymbol{u}$.

We are now ready to describe the proposed algorithm *AlgMaxNoEn* (Fig. 2). The intuitive idea of the algorithm is to produce candidate solutions (i.e., fragments) and verify whether they are actual solutions, that is, if they are maximally X-incomplete fragments. Candidate solutions are produced by incrementally removing propositions from the set $\mathcal{P}$, which from now on we assume does not contain X (as X cannot occur in any solution). Once a proposition is removed, the status of the resulting fragment is checked: if it is maximally X-incomplete, then it is added to the solution set; otherwise, either the computation continues by refining the candidate solution through the removal of another proposition or, if refining this candidate is considered not promising (according to criteria that will be defined later on), the analysis of this candidate ends and we focus on a new candidate.

The process is carried out in a recursive fashion, *AlgMaxNoEnR* being the recursive function and *AlgMaxNoEn* being the wrapper function, which executes the first call to *AlgMaxNoEnR* (see Fig. 2). The parameters of the recursion are

the fragment F, representing the propositions that have been already removed (thus the candidate under analysis is its complement $\mathcal{P} \setminus F$), and the fragment V, which is a (not necessarily strict) subset of $\mathcal{P} \setminus F$ and represents the propositions that can still be removed to refine the current candidate. (The additional parameters of *AlgMaxNoEnR* can be thought of as global variables, as they are not involved in the recursion process: $\mathcal{L}$ collects the solutions, while $\mathcal{P}$, $\mathcal{K}$, and X represent the instance given as input to *AlgMaxNoEn*.) Thus, a generic recursive call on F and V analyses, as a candidate, the complement of F, which can be possibly refined, in successive recursive calls, through the removal of (some of) the propositions in V. In this way, the recursive function searches for solutions contained in the whole set of sub-fragment of $\mathcal{P} \setminus F$.

Given an instance $\langle \mathcal{P}, \mathcal{K}, X \rangle$ of MaxNoEntail as input, the wrapper function *AlgMaxNoEn* (Fig. 2, left-hand side, top) executes the first recursive call to *AlgMaxNoEnR* on the recursive parameters $F = \emptyset$ and $V = \mathcal{P}$. The function *AlgMaxNoEnR* recursively builds a tree isomorphic to its own recursion tree. Such a structure is actually useless for the purposes of the algorithm, but it will be handy for the correctness analysis. In what follows, nodes of the above-mentioned tree are identified by the pair $\langle F, V \rangle$ of recursive parameters on which the call is performed. Thus, there is a one-to-one correspondence between nodes and calls to *AlgMaxNoEnR*. For the sake of simplicity, we will sometimes refer to a call to *AlgMaxNoEnR* through its corresponding node, and vice versa. For example, we will say that "a node $\mathcal{N}$ returns the exit-value r", meaning that the corresponding call returns r. A call to *AlgMaxNoEnR* may produce one of three outcomes: 'solution found', 'no solution', or 'non-maximally incomplete'. Intuitively, the value 'solution found' is returned by a node when a solution has been found in its own sub-tree (i.e., in the sub-tree rooted at it); if this is the case, we also say that the node *sees* a solution. Otherwise, if a fragment that is maximally X-incomplete in $\mathcal{K}$ has been analysed in its own sub-tree, the value 'non-maximally incomplete' is returned. The value 'no solution' is returned when none of the two cases above applies.

As a first step, the algorithm checks if one of the base-case conditions is met. Clearly, if a base-case condition is met inside a call, then that call corresponds to a leaf of the recursion tree. Base-case conditions allow the algorithm to end the analysis of a candidate, with no further refinements (and thus avoiding the exploration of the set of its sub-fragments), because either the candidate itself is a solution or its refinement is not promising. The refinement of a solution $\mathcal{P} \setminus F$ by removing propositions in V is not promising when the corresponding node $\langle F, V \rangle$ does not see any solution. Clearly, this is the case when $\mathcal{P} \setminus (F \cup V) \models_{\mathcal{K}} X$: if the weakest fragment $\mathcal{P} \setminus (F \cup V)$ of the set of sub-fragments of $\mathcal{P} \setminus F$ entails X, then, due to the monotonicity of entailment, all the fragments of the set do, meaning that none of them is X-incomplete. Another case in which refining a candidate is not promising is when the candidate F is X-incomplete but not maximally X-incomplete: if F is not maximally X-incomplete, then all its sub-fragments are not, either. We are interested in detecting such non-promising situations as soon as possible, to reduce the number of candidates analysed by the algorithm.

To this end, we use the above-mentioned property that the complement of a maximally X-incomplete fragment is useful (Lemma 4). This implies that, if a fragment F is not useful, that is, some of its propositions do not occur in its utility vector, then neither $\mathcal{P} \setminus F$ nor any of its sub-fragments is a solution, and thus the analysis of $\mathcal{P} \setminus F$ ends with no further refinement. Thus, there are three base-case conditions. *(i)* If *compUtilityVec* (see Fig. 2, left-hand side, bottom) returns `false`, then F is not useful, and the function returns 'non-maximally incomplete'. *(ii)* If $\mathcal{P} \setminus (F \cup V)$ entails X, then refining $\mathcal{P} \setminus F$ cannot lead to a solution, and the function returns 'no solution'. *(iii)* If the complement of F is X-incomplete, then it may be a solution. Its maximality is checked, exploiting the monotonicity of entailment, and, depending on the result of this test, either it is added to the solution set $\mathcal{L}$ and 'solution found' is returned, or 'non-maximally incomplete' is returned.

If none of the above base-case conditions is met, the refinement of the candidate is performed (`while` loop in *AlgMaxNoEnR*, in Fig. 2). At each iteration, an element Y of V is selected. Each iteration corresponds to an attempt to extend F with the new proposition Y and the new node $\langle F \cup \{Y\}, V \setminus \{Y\}\rangle$ (corresponding to the recursive call on $F \cup \{Y\}$ and $V \setminus \{Y\}$) is created as a child of the current one. Depending on the value returned from a recursive call, the local variables *flagSol*, *flagNoMax*, and *keep* are suitably updated. Intuitively, *FlagSol* is `true` if and only if the current node sees a solution. If the current node sees no solutions, but at least one of the nodes in its own sub-tree returned 'non-maximally incomplete', then *flagNoMax* is `true`. Finally, *keep* is set to `false` as soon as a call returns 'no solution'. In this last case, thanks to the monotonicity of the entailment, we can exit the current loop, as no other solution can be produced by refining the current candidate $\mathcal{P} \setminus F$. The return value after the loop is then returned depending on the values of *flagSol* and *flagNoMax*.

In the what follows, let $\mathcal{T}$ be the tree rooted at the node $\langle \emptyset, \mathcal{P}\rangle$, as generated by the first call to the recursive function *AlgMaxNoEn*. The following theorem states that the proposed algorithm is sound and complete.

Theorem 2. *Let $\langle \mathcal{P}, \mathcal{K}, X\rangle$ be an instance of* MaxNoEntail. *Then, a fragment is included in the set of solutions returned by the algorithm AlgMaxNoEn on input $\langle \mathcal{P}, \mathcal{K}, X\rangle$ if and only if it is maximally X-incomplete in $\mathcal{K}$.*

Proof. The soundness of *AlgMaxNoEn* follows from the description of the algorithm: a fragment is included in the set of solutions returned by *AlgMaxNoEn* only if the test for its maximal X-incompleteness succeeds.

To prove the completeness of *AlgMaxNoEn*, let us consider a generic maximally X-incomplete fragment F. We show that a node $\mathcal{N} = \langle \overline{F}, V\rangle$, where $\overline{F} = \mathcal{P} \setminus F$, is eventually created and processed, for some $V \subseteq F$. As F is indeed a maximally X-incomplete fragment, the corresponding base-case condition applies, and F is added to the solution set. Let us consider the ordering over $\mathcal{P}$ according to which the elements are selected inside the `while` loop of the algorithm *AlgMaxNoEnR* (see Fig. 2). Let A be the first occurrence in $\mathcal{P}$ of an element of $\overline{F}$ and let $B_1, \ldots, B_s$ be the elements preceding A in $\mathcal{P}$ (according to

the above-mentioned ordering). We have to show that the child $\mathcal{N}_A = \langle\{A\}, V_A\rangle$, where $V_A = (\mathcal{P} \setminus \{A\}) \setminus \{B_i \mid 1 \leq i \leq s\}$, is eventually processed. Notice that $\mathcal{P} \setminus (V_A \cup \{A\}) \subseteq F$ holds, as $\overline{F} \subseteq V_A \cup \{A\}$. Thus, $\mathcal{P} \setminus (V_A \cup \{A\})$ is X-incomplete, as well (as F is and by monotonicity of the entailment). Suppose, towards a contradiction, that $\mathcal{N}_A$ is never processed. Then, one of its left siblings $\mathcal{N}_{B_i} = \langle\{B_i\}, V_{B_i}\rangle$, for some $i \in \{1, \ldots, s\}$, where $V_{B_i} = \mathcal{P} \setminus \{B_j \mid 1 \leq j \leq i\}$, has returned 'no solution'. Since $\mathcal{P} \setminus (V_A \cup \{A\})$ is X-incomplete and $V_A \cup \{A\} \subseteq V_{B_i}$ holds, there exists a path from $\mathcal{N}_{B_i}$ to a leaf corresponding to a candidate that is X-incomplete. Such a leaf returns either 'solution found' or 'non-maximally incomplete', and thus $\mathcal{N}_{B_i}$ does not return 'no solution', leading to contradiction. As the same argument can be iterated for every other element of $\overline{F}$, we can conclude that $\mathcal{N}_A$ is processed, and we are done. □

5 Applications and experimental results

The algorithm for MaxNoEntail given in Section 4 outperforms brute-force ones given in Fig. 1. Moreover, thanks to the reduction provided in Section 3, it can also be exploited to solve MaxMod. In this section, we show a further application of *AlgMaxNoEn* as a tool to compare the expressive power of modal logics (see Section 1).

Given a set of modalities, an *inter-definability* (among them) describes how to define a modality in terms of others. An inter-definability can be thought of as a clause of a positive Horn theory (where atomic propositions play the role of the modalities). Consequently, a set of inter-definabilities is nothing but a positive Horn theory. Thus, the task of identifying the maximal subsets of modalities that, within a given set of inter-definabilities, do not express a specific modality amounts to solving MaxNoEntail. Actually, it was this very problem that motivated us to carry out this study, in the search for a better solution than the one based on the brute-force approach. While in modal logics with few operators and inter-definabilities, the above-mentioned task can be easily carried out by hand (as it has been done, e.g., in [9]), in modal logics with many operators and several inter-definabilities, it may require a big and error-prone effort. Even though most modal logics have a small set of modalities, there are meaningful ones that feature tens of modalities (see, e.g., [2,10]). In [1], the authors proposed, and used, a naïve, brute-force algorithm similar to the ones presented in Fig. 1 to perform the aforementioned task. Even if this approach was efficient enough for the particular modal logic studied in [1], it turned out to be unsuitable to deal with logics with larger sets of modalities, such as the one studied in [2], featuring more than 20 modalities.

We have carried out an experimental comparison of the efficiency of the algorithm *AlgMaxNoEn* vis-a-vis those given in Fig. 1. We summarize the outcomes of our experiments[4] in Fig. 3. (A more detailed account is given in Table 1 in Appendix B.) For each pair of values n and k, ranging, respectively, between 12 and

[4] All the experiments were executed on a PC system with an Intel® Core™i3-2120 CPU @ 3.30GHz × 4 and 7.7 GB of RAM, under Ubuntu Linux 12.04 (precise) 64-

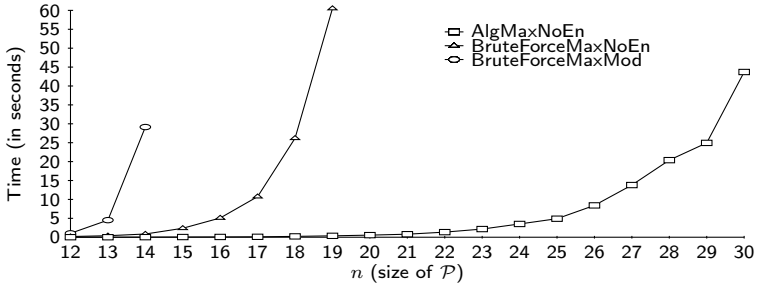

Fig. 3. Running times of the three algorithms on randomly-generated instances.

30 and between $\lfloor n/3 \rfloor$ and n, the running times of the three algorithms presented in this papers (i.e., *BruteForceMaxMod*, *BruteForceMaxNoEntail*, and *AlgMaxNoEn*) are compared with respect to a set of seven randomly-generated Horn theories $\mathcal{K}$ over $\mathcal{P}$, where $|\mathcal{P}| = n$ and $|\mathcal{K}| = k$ (to be precise, *BruteForceMaxMod* is run on randomly-generated instances of the form $\langle \mathcal{P}, \mathcal{K} \rangle$, while *BruteForceMaxNoEntail* and *AlgMaxNoEn* on instances of the form $\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle$, obtained from the instances used for testing *BruteForceMaxMod* through the reduction described in Section 3). The chart in Fig. 3 reports the average running times of the three algorithms for the different values of n (size of $\mathcal{P}$). In spite of a similar, exponential trend exhibited by the three algorithms (notice that such a behaviour is unavoidable as the problems can produce outputs whose size is, in general, exponential in the size of the input), our tests show that the two algorithms based on a brute-force approach become inefficient already for instances over set of propositions of size 15 and 20, respectively, and are thus unable to deal, for instance, with the logic studied in [2]. On the other hand, *AlgMaxNoEn* can deal with all tested instances in reasonable time.

6 Conclusions

In this paper we have studied the problem of enumerating the maximal models of a Horn theory (MaxMod) and we established a connection between this problem and the problem of enumerating the maximal subsets of facts that do not entail a distinguished atomic proposition in a positive Horn theory (MaxNoEntail). We first showed that the two problems are polynomially equivalent and then we presented an algorithm for MaxNoEntail that performs better than the ones based on a brute-force approach. As the problems can produce an output of size, in general, exponential in the size of the input, it not possible to avoid the exponential trend shown by the algorithms in Fig. 3. Moreover, in [12], it has been proved that, unless P=NP, no output-polynomial algorithm can be devised for MaxMod (and thus for MaxNoEntail), meaning that it is not

bit. On the web-page `http://www.di.unisa.it/dottorandi/dario.dellamonica/download/lpar13_code.zip` it is possible to download the source code in C++.

even possible to devise an algorithm that runs in polynomial time in terms of both the sizes of input and output. Nevertheless, our approach is efficient enough to allow us to deal with some expressiveness issues for modal logics that were intractable with the brute-force approach, as shown by empirical evidence.

The proposed algorithm can be improved by conceiving suitable heuristics to drive the construction of the candidate solution (e.g, heuristics for the choice of the next atomic proposition to be removed from the fragment) and by suitably reducing, on the fly, the Horn theory depending on the current candidate under analysis. We plan to explore both such possibilities in future work. We also intend to investigate the behaviour of the proposed algorithm on special instances of MaxNoEntail, i.e., on Horn theories whose clauses have the same consequent. Such a restriction makes the MaxNoEntail problem equivalent to the well-known problem of finding the minimal hitting sets of a hyper-graph, for which it is still an open question whether an output-polynomial algorithm exists.

References

1. L. Aceto, D. Della Monica, A. Ingólfsdóttir, A. Montanari, and G. Sciavicco. A complete classification of the expressiveness of interval logics of Allen's relations over dense linear orders. In *Proc. of the 20th TIME*, 2013.
2. P. Balbiani, V. Goranko, and G. Sciavicco. Two-sorted point-interval temporal logics. *Electr. Notes Theor. Comput. Sci.*, 278:31–45, 2011.
3. P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 2002.
4. R. Brachman and H. Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers Inc., 2004.
5. V. Brusoni, L. Console, P. Terenziani, and D. Theseider Dupré. Characterizing temporal abductive diagnosis. In *Proc. of the 6th DX*, pages 34–40, 1995.
6. M. Cadoli. The complexity of model checking for circumscriptive formulae. *Information Processing Letters*, 44:113–118, 1992.
7. C. Chang and R.C. Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, Inc., 1st edition, 1997.
8. S. Cook. The complexity of theorem proving procedures. In *Proc. of the 3rd Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.
9. D. Della Monica, V. Goranko, A. Montanari, and G. Sciavicco. Expressiveness of the interval logics of Allen's relations on the class of all linear orders: Complete classification. In *Proc. of the 22nd IJCAI*, pages 845–850, 2011.
10. J. Halpern and Y. Shoham. A propositional modal logic of time intervals. *Journal of the ACM*, 38(4):935–962, 1991.
11. H. Kautz, D. Mcallester, and B. Selman. Encoding plans in propositional logic. In *Proc. of the 5th KR*, pages 374–384, 1996.
12. D. J. Kavvadias, M. Sideri, and E.C. Stavropoulos. Generating all maximal models of a Boolean expression. *Inf. Process. Lett.*, 74(3–4):157–162, 2000.
13. D.E. Knuth. *The Art of Computer Programming: Combinatorial Algorithms, Part 1*, volume 4A. Addison-Wesley Professional, 1st edition, 2011.
14. J.W. Lloyd. *Foundations of Logic Programming*. Springer, 2nd edition, 1987.
15. C.K. Riesbeck and R.C. Schank. *Inside Case-based Reasoning*. Artificial intelligence series. Lawrence Erlbaum, 1989.
16. J. Schmidt. Enumeration: Algorithms and complexity. Unpublished, 2009.

Appendix

A Proof of Lemma 1

Proof. (a) Let F be a fragment of $\mathcal{P}$ that is maximally X-incomplete in $\mathcal{K}$. If $A \in F$, then $F \cup \{\neg A\}$ is unsatisfiable, and therefore $F \models_{\mathcal{K}} A$ follows by the definition of entailment. To prove the converse implication, let us suppose, for the sake of contradiction, that $F \models_{\mathcal{K}} A$ and $A \notin F$. By the definition of entailment, it follows that $\mathcal{K} \cup F \cup \{\neg A\}$ is unsatisfiable, that is, every model M of $\mathcal{K} \cup F$ is such that $M(A) = 1$. Since F is X-incomplete, $F \not\models_{\mathcal{K}} X$ holds, which means that $\mathcal{K} \cup F \cup \{\neg X\}$ is satisfiable. Now, consider a model M that satisfies $\mathcal{K} \cup F \cup \{\neg X\}$. Clearly, it satisfies $\mathcal{K} \cup F$, as well. Thus, we have that $M(A) = 1$. Then, $\mathcal{K} \cup F \cup \{A\} \cup \{\neg X\}$ is satisfiable, which implies $F \cup \{A\} \not\models_{\mathcal{K}} X$, contradicting the assumption that F is maximally X-incomplete.

(b) Let M be a model of $\mathcal{K}$. If $M(A) = 1$, then $A \in \eta(M)$, which, in turn, implies $\eta(M) \models_{\mathcal{K}} A$, and thus $M \models_{\mathcal{K}} A$. To prove the converse implication, let us assume that $M \models_{\mathcal{K}} A$. By the definition of entailment, $\eta(M) \cup \mathcal{K} \cup \{\neg A\}$ is unsatisfiable. This means that each model of $\mathcal{K} \cup \eta(M)$ is such that $M(A) = 1$. Since M is a model of $\mathcal{K}$ (by our assumption) and M is a model of $\eta(M)$ (by the definition of $\eta(M)$), it follows that $M(A) = 1$, which was to be shown. □

B Detailed account of the experiments

The following table contains a detailed account of our tests. The parameter n corresponds to the size of the set of atomic propositions $\mathcal{P}$ and ranges between 12 and 30. The parameter k corresponds to the number of clauses of the Horn theory and ranges between $\lfloor n/3 \rfloor$ and n. For each pair of values n and k, seven instances $\langle \mathcal{P}, \mathcal{K} \rangle$ of MaxMod are randomly generated and the algorithm *BruteForceMaxMod* is run on each of them. Then, each of such instances is transformed into an instance of MaxNoEntail of the form $\langle \mathcal{P} \cup \{X\}, \tau(\mathcal{K}), X \rangle$, by applying the reduction described in Section 3. On this latter set of instances, the algorithms *BruteForceMaxNoEntail* and *AlgMaxNoEn* are executed. The table reports, for each pair of values n and k, the average running times, in seconds and rounded to the third decimal digit, of the three algorithms on the set of randomly-generated instances. Moreover, the last column shows the average running times of the algorithms for any fixed value of n (this values are the ones used for the chart in Fig. 3). In each entry of the table, the first (resp., second, third) row corresponds to the average running time of the algorithm *BruteForceMaxMod* (resp., *BruteForceMaxNoEntail*, *AlgMaxNoEn*). A blank entry of the table means that no instances were generated for that pair of values n and k. An hyphen in one of the entries means that the corresponding algorithm timed out, that is, it ran for more than 30 minutes, on that set of instances.

Table 1. Experimental results.

n \ k	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	Av.
12	0.091 0.014 0	0.201 0.034 0.003	1.157 0.096 0.001	1.897 0.156 0.009	0.736 0.136 0.007	1.483 0.253 0.011	1.113 0.283 0.017	0.871 0.317 0.013	1.75 0.404 0.026																			1.033 0.188 0.01
13	1.053 0.044 0.006	5.646 0.146 0.003	4.506 0.144 0.007	3.523 0.213 0.006	1.714 0.28 0.007	5.881 0.49 0.014	6.781 0.599 0.017	8.277 0.663 0.02	4.361 0.653 0.031	3.037 0.886 0.03																		4.478 0.412 0.014
14	46.259 0.21 0.004	12.861 0.231 0	32.601 0.284 0.004	46.601 0.5 0.009	33.261 0.689 0.011	7.029 0.673 0.011	28.457 0.973 0.037	48.509 1.329 0.041	26.52 1.386 0.06	11.683 1.577 0.047	27.889 2.001 0.1																	29.243 0.896 0.029
15		7.717 0.276 0.003	92.466 0.704 0.007	– 1.145 0.025	– 1.593 0.039	– 1.969 0.036	– 2.296 0.036	– 2.237 0.034	– 3.314 0.104	– 3.771 0.091	– 3.617 0.089	– 4.001 0.104																– 2.266 0.052
16		– 0.51 0.006	– 0.451 0.009	– 1.089 0.006	– 1.864 0.021	– 4.866 0.029	– 4.537 0.069	– 6.233 0.084	– 5.813 0.086	– 6.953 0.1	– 9.323 0.18	– 8.94 0.187	– 10.519 0.151															– 5.092 0.077
17		– 0.62 0.006	– 2.049 0.013	– 2.51 0.019	– 7.33 0.037	– 4.931 0.036	– 12.427 0.09	– 9.54 0.076	– 14.521 0.1	– 12.52 0.12	– 14.836 0.247	– 17.493 0.254	– 18.064 0.266	– 22.849 0.204														– 10.745 0.113
18			– 6.607 0.013	– 9.281 0.031	– 9.441 0.034	– 16.371 0.067	– 13.419 0.094	– 17.793 0.111	– 27.489 0.173	– 29.349 0.237	– 30.16 0.176	– 37.143 0.41	– 36.42 0.549	– 50.483 0.529	– 57.163 0.531												– 26.240 0.227
19			– 8.284 0.036	– 35.89 0.071	– 16.207 0.079	– 30.613 0.074	– 39.523 0.129	– 40.76 0.179	– 47.316 0.207	– 57.939 0.171	– 79.539 0.477	– 78.7 0.569	– 88.126 0.591	– 102.786 0.811	– 106.2 0.674	– 115.561 1.221											– 60.532 0.378
20			– 22.497 0.03	– 18.966 0.051	– 19.474 0.051	– 77.469 0.109	– 62.401 0.156	– 142.477 0.237	– 95.103 0.397	– 178.64 0.493	– 105.016 0.473	– 176.824 0.517	– 187.417 0.613	– 219.826 1.356	– 207.391 0.867	– – 1.437	– – 1.549										– – 0.556
21				– – 0.056	– – 0.116	– – 0.111	– – 0.161	– – 0.334	– – 0.497	– – 0.461	– – 0.894	– – 1.043	– – 1.09	– – 0.934	– – 1.53	– – 1.444	– – 1.631	– – 2.52										– – 0.855
22				– – 0.036	– – 0.103	– – 0.189	– – 0.329	– – 0.296	– – 0.703	– – 0.664	– – 0.756	– – 1.32	– – 1.893	– – 1.346	– – 2.314	– – 2.374	– – 2.923	– – 3.02	– – 3.163								– – 1.339
23				– – 0.07	– – 0.137	– – 0.237	– – 0.394	– – 0.597	– – 0.72	– – 1.109	– – 1.286	– – 1.351	– – 2.074	– – 2.834	– – 3.621	– – 4.763	– – 5.021	– – 2.553	– – 6.366	– – 3.851								– – 2.176
24					– – 0.144	– – 0.39	– – 0.44	– – 0.886	– – 0.694	– – 1.491	– – 1.3	– – 2.764	– – 2.117	– – 4.21	– – 4.997	– – 5.25	– – 5.796	– – 8.616	– – 4.47	– – 7.061	– – 9.249							– – 3.522
25					– – 0.196	– – 0.42	– – 0.299	– – 0.591	– – 0.767	– – 0.761	– – 2.267	– – 3.261	– – 4.53	– – 3.729	– – 4.606	– – 3.73	– – 4.629	– – 9.453	– – 9.151	– – 13.08	– – 15.783	– – 11.114						– – 4.909
26					– – 0.336	– – 0.241	– – 0.474	– – 0.874	– – 1.044	– – 1.296	– – 2.463	– – 3.557	– – 4.676	– – 7.714	– – 6.089	– – 5.989	– – 12.301	– – 13.603	– – 11.357	– – 16.207	– – 16.023	– – 20.811	– – 34.237					– – 8.384
27						– – 0.5	– – 0.514	– – 1.151	– – 1.493	– – 2.274	– – 3.39	– – 1.874	– – 5.346	– – 6.059	– – 15.029	– – 13.17	– – 15.749	– – 10.707	– – 25.574	– – 28.521	– – 26.951	– – 28.251	– – 33.033	– – 31.801				– – 13.231
28						– – 0.466	– – 0.857	– – 1.036	– – 2.03	– – 4.541	– – 5.274	– – 4.104	– – 5.389	– – 10.757	– – 5.233	– – 18.759	– – 23.093	– – 27.074	– – 20.263	– – 16.876	– – 24.347	– – 68.16	– – 58.693	– – 48.851	– – 62.25			– – 20.403
29						– – 0.46	– – 0.999	– – 1.729	– – 2.896	– – 4.46	– – 5.827	– – 7.89	– – 10.484	– – 19.524	– – 15.26	– – 26.79	– – 17.574	– – 20.667	– – 24.231	– – 37.986	– – 49.063	– – 19.827	– – 36.014	– – 20.529	– – 129.76	– – 72.223		– – 24.962
30							– – 1.466	– – 2.297	– – 2.77	– – 3.133	– – 7.066	– – 7.74	– – 10.507	– – 10.17	– – 15.059	– – 17.9	– – 22.519	– – 20.126	– – 34.514	– – 41.966	– – 41.441	– – 46.057	– – 131.967	– – 112.599	– – 145.011	– – 121.006	– – 122.6	– – 43.710