A simple functional presentation and an inductive correctness proof of the Horn algorithm

António Ravara
NOVA-LINCS and Dep. de Informática, FCT.
Universidade NOVA de Lisboa, Portugal

We present a recursive formulation of the Horn algorithm for deciding the satisfiability of propositional clauses. The usual presentations in imperative pseudo-code are informal and not suitable for simple proofs of its main properties. By defining the algorithm as a recursive function (computing a least fixed-point), we achieve: 1) a concise, yet rigorous, formalisation; 2) a clear form of visualising executions of the algorithm, step-by-step; 3) precise results, simple to state and with clean inductive proofs.

1 Motivation

The Horn algorithm [Hor51] is a particularly efficient decision procedure for the satisfiability problem of propositional logic. Although Horn Clause Logic is computationally complete, the satisfiability problem for the conjunction of Horn clauses is **P**-complete and nevertheless provable in linear time (there is an algorithm that takes at most n steps to determine if the conjunction of Horn clauses is satisfiable) [CN10, DG84]. Note that the general Boolean satisfiability problem (for arbitrary propositional formulae) is **NP**-complete.

Textbooks on (Mathematical or Computational) Logic usually present imperative formulations of this algorithm, with rather informal proof sketches [Hed04, HR04]. To present a correctness proof in full detail, one would need to follow, for instance, the Hoare style, defining the syntax of the programming and of an assertion languages, the operational semantics, the proof system (at least discussing its correctness), and then present the axiomatic proof. The setting is a bit demanding and requires some auxiliar "machinery".

We believe a formulation of the algorithm as a recursive function allows for not only a simple and easily readable definition, but mainly, allows for a simple (inductive) proof, which in turn sheds light on the algorithm itself, leading to several possible improvements.

We present herein such a formulation together with examples of execution, a correctness proof, and some further results useful for optimisations of the algorithm.

2 The Horn algorithm

Motivation. If a propositional formula φ is in Conjunctive Normal Form (or $\mathsf{CNF}(\varphi)$, according to Definition A.20), then checking that φ is valid is straightforward: it has polynomial complexity (with respect to the number of propositional symbols occurring in the formula). The Horn algorithm is a simple and fast solution (polynomial as well) to determine if a formula is satisfiable or contradictory. However, the algorithm works only for a certain class of formulae — the *Horn Clauses*.

Submitted to:
HCVS 2018

Syntax. Let P be a countable set of propositional symbols, ranged over by $p,q,\ldots$, and consider $Alf_P = P\cup\{\bot,\vee,\wedge,\rightarrow,(,)\}$ a propositional alphabet over P. The set F_P of propositional formulæ is the least one including the symbols in P, the symbol $\bot$, and closed for the operators $\vee,\wedge,\rightarrow$ (*cf.* Definitions A.2 and A.3).

2.1 Horn Clauses

Recall that a literal is an atomic formula or its negation (*cf.* Definition A.18).

Definition 2.1. *A* basic Horn clause *is a disjunction of literals where at most one occurs positively.*

Formulæ like $\bot$, p, $p\vee\neg q$, and $\neg p\vee\neg q$ are basic Horn clauses, whereas $p\vee q$ or $\bot\vee p$ are not.

Horn formulæ. Note that a basic Horn clause is in one of the following three cases: (1) does not have positive literals; (2) does not have negative literals (and so it is a single positive literal); (3) it has negative literals and one positive. Therefore, any basic Horn clause may be presented as an implication. Let '$\equiv$' stand for logical equivalence (*cf.* Definition A.16).

Lemma 2.2. *Let L and L_i (for all considered i) be positive literals.*

1. $L\equiv\top\rightarrow L$
2. $\bigvee_{i=1}^{n}\neg L_i\equiv(\bigwedge_{i=1}^{n}L_i)\rightarrow\bot$
3. $\bigvee_{i=1}^{n}\neg L_i\vee L\equiv(\bigwedge_{i=1}^{n}L_i)\rightarrow L$

Proof. In Appendix B. □

We define now when is a propositional formula a Horn clause.

Definition 2.3. *A formula $\varphi\in F_P$ such that* CNF*(φ) is a* Horn clause*, if it is the conjunction of basic Horn clauses.*

Let E_P denote the set of propositional formulæ obtained by considering negation a primitive operator.

Proposition 2.4. *Let $\varphi\in E_P$ be a Horn clause; then, $\varphi\equiv\bigwedge_{i=1}^{n}(C_i\rightarrow L_i)$, for some $n\geq 1$, where, for any $i\in\{1,\ldots,n\}$, each L_i is a positive literal, each $C_i=\top$ or $C_i=\bigwedge_{j=1}^{k_i}L_{i,j}$, with $k_i\geq 1$, and each $L_{i,j}$ is a positive literal.*

Proof. Use the previous lemma to transform each basic clause in an implication. □

Henceforth, we call *Horn formula* to a Horn clause $\varphi\in E_P$ such that

$$\varphi=\bigwedge_{i=1}^{n}(C_i\rightarrow L_i)$$

2.2 A functional presentation of the algorithm

The main contribution of this note is the (non-deterministic, for simplicity)[1] recursive formulation of the Horn algorithm, together with the proof of correctness and the optimisation lemmas.

[1] A deterministic formulation is achieved easily, *e.g.* by inspecting the formula from left to right.

Definition 2.5. *Let φ be a Horn formula. We define the function $\mathscr{H} : E_P \to \{0,1\}$ as*

$$\mathscr{H}(\varphi) = \begin{cases} 1, & \text{if } \perp \notin \mathscr{A}(\varphi, \{\top\}) \\ 0, & \text{otherwise} \end{cases}$$

with $\mathscr{A} : E_P \times \wp(\{\perp,\top\} \cup P) \to \wp(\{\perp,\top\} \cup P)$ being the following function over Horn formulæ.

$$\mathscr{A}(\varphi,\mathscr{C}) = \top \text{ if } \varphi \equiv \top;\ \text{otherwise } \mathscr{A}(\varphi,\mathscr{C}) = \begin{cases} \mathscr{A}(\varphi \setminus (C_i \to L_i), \mathscr{C} \cup \{L_i\}), & \text{if } \exists_{i\in\{1,\dots,n\}}.\, \mathsf{set}(C_i) \subseteq \mathscr{C} \\ \mathscr{C}, & \text{otherwise} \end{cases}$$

where $\mathsf{set}(\top) = \{\top\}$ and $\mathsf{set}(\bigwedge_{i=1}^{k} L_i) = \{L_i \mid i \in \{1,\dots,k\}$ with $k \geq 1\}$; moreover, $\varphi \setminus \varphi \stackrel{\text{def}}{=} \top$ and $\varphi \setminus (C_i \to L_i) \stackrel{\text{def}}{=} (\bigwedge_{j=1}^{i-1}(C_j \to L_j)) \wedge (\bigwedge_{j=i+1}^{n}(C_j \to L_j))$, if $i > 1$.

To illustrate how the algorithm works, we present some representative examples. Let us first state the main property of the algorithm. Recall that a formula is satisfiable if it is satisfied by some valuation and is contradictory if no valuation satisfies it (*cf.* Definition A.10 and subsequent lemmas).

Theorem 2.6. *For any Horn clause $\varphi \in E_P$:*

- $\mathscr{H}(\varphi) = 1$ *if, and only if, φ it is* satisfiable*;*
- $\mathscr{H}(\varphi) = 0$ *if, and only if, φ it is* contradictory.

Proof. A consequence of Theorem 3.6 (presented ahead). □

Example 2.7. *Let us determine the nature of the following Horn clause.*

$$\varphi \stackrel{\text{def}}{=} p \wedge (\neg r \vee s) \wedge (r \vee \neg p \vee \neg q) \wedge (\neg r \vee \neg s) \wedge q$$

Notice that φ is a CNF*, but (according to Lemma A.19) it is not valid. We convert it to a Horn formula using Lemma 2.2.*

$$\varphi \equiv \psi \stackrel{\text{def}}{=} (\top \to p) \wedge (r \to s) \wedge ((p \wedge q) \to r) \wedge ((r \wedge s) \to \perp) \wedge (\top \to q)$$

Considering

$$\begin{aligned} \psi_1 &= (r \to s) \wedge ((p \wedge q) \to r) \wedge ((r \wedge s) \to \perp) \wedge (\top \to q) \\ \psi_2 &= (r \to s) \wedge ((p \wedge q) \to r) \wedge ((r \wedge s) \to \perp) \end{aligned}$$

we calculate the function $\mathscr{A}$.

$$\begin{aligned} \mathscr{A}(\psi, \{\top\}) &= \mathscr{A}(\psi_1, \{\top, p\}) \\ &= \mathscr{A}(\psi_2, \{\top, p, q\}) \\ &= \mathscr{A}((r \to s) \wedge ((r \wedge s) \to \perp), \{\top, p, q, r\}) \\ &= \mathscr{A}((r \wedge s) \to \perp, \{\top, p, q, r, s\}) \\ &= \mathscr{A}(\top, \{\top, p, q, r, s, \perp\}) \\ &= \{\top, p, q, r, s, \perp\} \end{aligned}$$

Since $\perp \in \{\top, p, q, r, s, \perp\}$, then $\mathscr{H}(\psi) = 0$; therefore ψ is contradictory, and since $\varphi \equiv \psi$, so is φ.

Example 2.8. *Let us now determine the nature of the following Horn clause.*

$$\varphi \stackrel{\text{def}}{=} p \wedge (\neg r \vee s) \wedge (r \vee \neg p \vee \neg q) \wedge (\neg r \vee \neg s)$$

Notice that φ *is a* CNF, *but (according to Lemma A.19) it is not valid. We convert it to a Horn formula*

$$\varphi \equiv \psi \stackrel{\text{def}}{=} (\top \to p) \wedge (r \to s) \wedge ((p \wedge q) \to r) \wedge ((r \wedge s) \to \bot)$$

and considering

$$\psi_1 = (r \to s) \wedge ((p \wedge q) \to r) \wedge ((r \wedge s) \to \bot))$$

we calculate the function $\mathscr{A}$.

$$\begin{array}{rl} \mathscr{A}(\psi, \{\top\}) & = \\ \mathscr{A}(\psi_1, \{\top, p\}) & = \\ \{\top, p\} & \end{array}$$

Since $\bot \notin \{\top, p\}$, *then* $\mathscr{H}(\psi) = 1$*; therefore* ψ *is satisfiable, and since* $\varphi \equiv \psi$*, so is* φ.

Indeed, considering V *where* $V(p) = 1$ *and* $V(q) = V(r) = V(s) = 0$*, one easily verifies that* V *satisfies* φ.[2]

Example 2.9. *Let us finally determine the nature of the Horn clause* $p \wedge (\neg r \vee s) \wedge (r \vee \neg p) \wedge \neg r$. *Notice that it is a not valid* CNF *(according to Lemma A.19); we convert it to a Horn formula and considering*

$$\begin{aligned} \varphi &= (\top \to p) \wedge (r \to s) \wedge (p \to r) \wedge (r \to \bot) \\ \varphi_1 &= (r \to s) \wedge (p \to r) \wedge (r \to \bot) \\ \varphi_2 &= (r \to s) \wedge (r \to \bot) \end{aligned}$$

we calculate the function $\mathscr{A}$*, taking advantage of its monotonicity (cf. Lemma 1).*

$$\begin{array}{rl} \mathscr{A}(\varphi, \{\top\}) & = \\ \mathscr{A}(\varphi_1, \{\top, p\}) & = \\ \mathscr{A}(\varphi_2, \{\top, p, r\}) & \supseteq \\ \{\top, p, r, \bot\} & \end{array}$$

Since $\bot \in \mathscr{A}(\varphi, \{\top\})$*, then* $\mathscr{H}(\varphi) = 0$*; therefore* φ *is contradictory; since it is equivalent to the original formula, that one is also contradictory.*

3 Results

We state herein several relevant properties of the algorithm, namely its characterisation as a least fixed-point and its correctness. Proofs are in the appendices.

[2] A property capturing this fact is stated as Proposition 3.5.

3.1 Fixed-points

Considering $\mathscr{L}$ to be the set of all literals, the set $\wp(\mathscr{L})$ is a complete lattice with respect to set inclusion. Since the function $\mathscr{A}$ is monotone (result stated below), by the Knaster-Tarski Theorem [Tar55], the function $\mathscr{A}$ has (unique) maximal and minimal fixed points. In fact, when applied to the set $\{\top\}$, the algorithm calculates a least fixed-point of $\mathscr{A}$ (the proof is in Appendix C).

Lemma 3.1. *Let $\varphi = \bigwedge_{i=1}^{n}(C_i \to L_i)$ be a Horn formula. The function $\mathscr{A}$ is:*

1. *increasing: $\mathscr{C} \subseteq \mathscr{A}(\varphi,\mathscr{C}) \subseteq \mathscr{C} \cup \bigcup_{i=1}^{n}\{L_i\}$;*
2. *and monotone: if $\mathscr{C} \subseteq \mathscr{D}$ then $\mathscr{A}(\varphi,\mathscr{C}) \subseteq \mathscr{A}(\varphi,\mathscr{D})$.*

Notice that once an execution step of $\mathscr{A}$ adds a literal to the result set, that literal is never taken out. Therefore, once an execution step adds $\perp$ to the result set, the procedure may stop as $\perp$ shall necessarily be in the final set. Moreover, the least result set of the algorithm is the single set $\{\top\}$, the literal $\top$ is in all result sets, and the greatest one is composed by $\top$ and all the literals that appear in the consequence of the implications constituting the input Horn formula.

3.2 Auxiliary and optimization lemmas

We present a couple of (straightforward) results that allow, in some particular cases, for better performance of the algorithm. Notice that if $\perp$ is not in the consequent of an implication of a Horn formula φ, or if no antecedent is $\top$, then $\perp$ is not in $\mathscr{A}(\varphi,\{\top\})$. Then, φ is satisfiable (and one does not even need to execute the algorithm). The fact is a particular case of the following corollary of the previous lemma (it is the contra-positive of Lemma 3.1.1).

Corollary 3.2. *Let $\varphi = \bigwedge_{i=1}^{n}(C_i \to L_i)$ be a Horn formula. If $L \notin \bigcup_{i=1}^{n} L_i$ then $L \notin \mathscr{A}(\varphi,\{\top\})$.*

Furthermore, if there are no “unit clauses” (of the form $\top \to p$), the execution of the algorithm ends in one step, not modifying the initial set. The lemma below, a simple consequence of the definition of the algorithm, captures this fact.

Lemma 3.3. *Let $\varphi = \bigwedge_{i=1}^{n}(C_i \to L_i)$ be a Horn formula. If $\forall i.1 \le i \le n \wedge C_i \ne \top$, then $\mathscr{A}(\varphi,\{\top\}) = \{\top\}$ and the execution of $\mathscr{A}$ takes exactly one step.*

Proof. By definition of the function $\mathscr{A}$ (in Definition 2.5), if $T \notin \bigcup_{1\le i\le n} C_i$ then $\mathscr{A}(\varphi,\{\top\}) = \{\top\}$, and $\mathscr{A}$ is calculated in exactly one step (applying the base case of its recursive definition). $\square$

3.3 Termination and complexity

The algorithm always produces a unique result set for a given input, *i.e.*, it is a *function*, and it always *terminates*; moreover, it is linear in the size of the formula, with each recursive step examining all the atomic symbols occurring in one of the clauses (which is then removed from the formula).

Theorem 3.4. *For any Horn formula $\varphi \in E_P$ there is a unique set $\mathscr{C}$ of literals such that $\mathscr{A}(\varphi,\{\top\}) = \mathscr{C}$. Furthermore, the procedure takes at most $n+1$ steps, where n is the number of clauses of φ.*

The proof of this result is in Appendix D (as the proof of Theorem D.1).

3.4 Correctness

Notice first that the result of the algorithm determines a *unique least model*: if the formula is satisfiable, then one gets a valuation satisfying it by assigning value 1 to the propositional symbols occurring in the resulting set. The other symbols occurring in the formula are set to 0. Let $\mathsf{SMB}(\varphi)$ denote the set of propositional symbols of a formula φ.

Proposition 3.5. *For any Horn formula $\varphi \in E_P$, let $\mathscr{A}(\varphi, \{\top\}) = \mathscr{C}$ and $\bot \notin \mathscr{C}$. Then, $V \Vdash \varphi$ considering V such that $V(p) = 1$ for each $p \in \mathscr{C}$ and $V(q) = 0$ for each $q \in (P \setminus \mathscr{C})$.*

The proof of this result is in Appendix E (as the proof of Proposition E.2).

We finally state the main result: the algorithm is sound and complete for Horn formulæ.

Theorem 3.6. *For any Horn formula $\varphi \in E_P$:*

- $\bot \notin \mathscr{A}(\varphi, \{\top\})$*, if, and only if, φ it is satisfiable;*
- $\bot \in \mathscr{A}(\varphi, \{\top\})$*, if, and only if, φ it is contradictory.*

The proof of this result is in Appendix F (as the proofs of Theorems F.2 and F.3).

4 Conclusions

We present herein a new formulation of the Horn algorithm for deciding the satisfiability problem of propositional logic. We define the procedure as a recursive function, instead of the usual imperative formulation in pseudo-code. This presentation has several advantages:

1. It is concise and readable, being at the same time rigorous;
2. allows for a simple presentation of "manual" executions of the algorithm, being usable in undergraduate logic courses;
3. has simple inductive proofs of soundness and completeness;
4. leads to optimization results, easy to state, prove, and implement.

We develop such a formulation and show examples of execution, a correctness proof and some further results useful for optimizations of the algorithm. Computing solutions for our recursive formulation of the lagorithm is akin to the fixed point (Knaster-Tarski) least Herbrand model construction.

Acknowledgements

This work was partially supported by NOVA LINCS grant UID/CEC/04516/2013.

References

[CLP00] René Cori, Daniel Lascar, and Donald H. Pelletier. *Mathematical logic: a course with exercises; Pt. 1: Propositional calculus, Boolean algebras, predicate calculus*. Oxford University Press, 2000.

[CN10] Stephen Cook and Phuong Nguyen. *Logical Foundations of Proof Complexity*. Cambridge University Press, 2010.

[DG84] William F. Dowling and Jean H. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *Journal of Logic Programming*, 1(3):267–284, 1984.

[Gal87] Jean H. Gallier. *Logic for Computer Science: Foundations of Automatic Theorem Proving*. Wiley, 1987.

[Hed04] Shawn Hedman. *A First Course in Logic: An Introduction to Model Theory, Proof Theory, Computability, and Complexity*. Oxford University Press, 2004.

[Hor51] Alfred Horn. On sentences which are true of direct unions of algebras. *Journal of Symbolic Logic*, 16(1):14–21, 1951.

[HR04] Michael Huth and Mark Ryan. *Logic in Computer Science - Modelling and Reasoning about Systems (2. ed.)*. Cambridge University Press, 2004.

[Tar55] Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5(2):285–309, 1955.

A The language of Propositional Logic

We make a brief presentation of the main concepts of Propositional Logic, to keep the paper self-contained. We define the syntax of the logic, a satisfaction relation, a notion of logical equivalence, and finally, a normal form. We omit the proofs of the results presented, which are standard and may be found in most textbooks (*cf.* [Gal87] or [HR04]).

A.1 Syntax

We inductively define the language with a minimal set of connectives, defining the other (redundant) ones as abbreviations.

Definition A.1. *Let P be a countable set (of propositional symbols). The* Propositional Alphabet over a set *P is the set $Alf_P = P \cup \{\bot, \vee, \wedge, \rightarrow, (,)\}$*

Definition A.2. *The* Propositional Language induced by *Alf_P is the set F_P, defined by the following grammar.*

$$\varphi, \psi ::= \bot \mid p \mid (\varphi \rightarrow \psi)$$

Elements of F_P are called *formulae*. Symbols in P and $\bot$ are *atomic* formulae.

Definition A.3. *The following* abbreviations *are useful.*

- *Negation: $\neg\varphi \stackrel{\text{abv}}{=} \varphi \rightarrow \bot$;*
- *Truth: $\top \stackrel{\text{abv}}{=} \neg\bot$;*
- *Disjunction: $\varphi \vee \psi \stackrel{\text{abv}}{=} \neg\varphi \rightarrow \psi$;*
- *Conjunction: $\varphi \wedge \psi \stackrel{\text{abv}}{=} \neg\varphi \vee \neg\psi$;*
- *Equivalence: $\varphi \leftrightarrow \psi \stackrel{\text{abv}}{=} (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$.*

Consider that the connective $\neg$ has precedence over all the other.

A.2 Semantics

We interpret the formulæ in a Boolean Algebra (like, *e.g.*, in [CLP00]).

Satisfaction relation.

Definition A.4. *A* valuation *over a set P of propositional symbols is a* function $V : P \rightarrow \{0,1\}$.

Definition A.5. *Consider the set $\{0,1\}$ equipped with two binary operations, $+$ and $\times$, interpreted as the addition and multiplication operations of the naturals, but such that $1+1=1$. An* interpretation function *of a formula $\varphi \in F_P$, for a given valuation V, denoted $[\![\varphi]\!]_V$, is a* function $[\![\cdot]\!]_V : F_P \rightarrow \{0,1\}$ *inductively defined by the following rules:*

- *$[\![p]\!]_V = V(p)$, for each $p \in P$;*
- *$[\![\bot]\!]_V = 0$;*
- *$[\![(\varphi \rightarrow \psi)]\!]_V = (1 - [\![\varphi]\!]_V) + [\![\psi]\!]_V$.*

Lemma A.6. *The following statements hold.*

- *$[\![(\varphi \vee \psi)]\!]_V = [\![\varphi]\!]_V + [\![\psi]\!]_V$;*

- $[\![(\varphi \wedge \psi)]\!]_V = [\![\varphi]\!]_V \times [\![\psi]\!]_V$.

Definition A.7. *Given a* valuation *V over P, the* satisfaction *of a formula $\varphi \in F_P$ by the valuation, denoted $V \Vdash \varphi$, is a relation containing the pair (V, φ), if $[\![\varphi]\!]_V = 1$.*

Hereafter we use the following terminology.

Definition A.8.
- *Whenever $V \Vdash \varphi$ one says that φ is* satisfied *by V.*
- *Whenever it is not the case that $V \Vdash \varphi$ (i.e., φ is not satisfied by V), one may write $V \nVdash \varphi$.*
- *Given $\Phi \subseteq F_P$, one may write $V \Vdash \Phi$, whenever $V \Vdash \varphi$ for each $\varphi \in \Phi$.*

Lemma A.9. *The following statements hold.*
- *$V \nVdash \varphi$ if, and only if, $V \Vdash \neg\varphi$*
- *$V \Vdash \varphi$ if, and only if, $V \nVdash \neg\varphi$*

Definition A.10. *A formula $\varphi \in F_P$ is:*
- satisfiable*, if $V \Vdash \varphi$, for some V;*
- valid *(denoted $\models \varphi$), if $V \Vdash \varphi$, for all V;*
- contradictory*, if no V is such that $V \Vdash \varphi$.*

One may write $\not\models \varphi$, if φ is *not* valid. The notion of satisfiability also applies to sets of formulae: a set $\Phi \subseteq F_P$ is *satisfiable*, if there is a V that satisfies every formula in Φ; otherwise, the set is said to be *contradictory*.

Lemma A.11. *A formula that is* not*:*
- *valid, is either satisfiable or contradictory;*
- *contradictory, is either satisfiable or valid;*
- *satisfiable, is contradictory (as it cannot be valid).*

Lemma A.12. *The* negation *of a formula:*
- *valid, is contradictory;*
- *contradictory, is valid;*
- *satisfiable (not valid), is satisfiable.*

Logical equivalence. There are syntactically different formulæthat evaluate to the same value, *i.e.*, are equivalent. To rigorously define the notion, we introduce first the idea a formula resulting from (or being a semantic consequence of) a set of formulæ.

Definition A.13. *Let $\Phi \subseteq F_P$ and $\varphi \in F_P$. One may say that a formula φ is a* semantic consequence *of a set of formulæ Φ, denoted by $\Phi \models \varphi$, if whenever $V \Vdash \Phi$ also $V \Vdash \varphi$.*

Proposition A.14. *$\{\varphi_1, \ldots, \varphi_n\} \models \psi$ if, and only if, $\models (\varphi_1 \wedge \ldots \wedge \varphi_n) \to \psi$, for any $n \in \mathbb{N}$.*

Lemma A.15. *The following statements hold.*
1. *$\{\perp\} \models \varphi$*
2. *$\{\varphi \wedge \psi\} \models \varphi$ and $\{\varphi \wedge \psi\} \models \psi$*
3. *$\{\varphi\} \models \varphi \vee \psi$ and $\{\psi\} \models \varphi \vee \psi$*

Definition A.16. *The formulae $\varphi, \psi \in F_P$ are* logically equivalent*, denoted by $\varphi \equiv \psi$, whenever $\{\varphi\} \models \psi$ if, and only if, $\{\psi\} \models \varphi$.*

Theorem A.17. *The binary relation $\equiv$ on F_P is a* congruence relation*.*

Conjunctive Normal Form.

Definition A.18. *A* literal *is an atomic formula (said* positive*) or the negation of an atomic formula (said* negative*).*

Recall that $\top \stackrel{\text{abv}}{=} \neg\bot$ (being thus a negative literal).

Lemma A.19. *A disjunction of literals $\bigvee_{i=1}^{n} L_i$, with $n \geq 1$, is valid if, and only if, there are $1 \leq i, j \leq n$ such that $L_i = \top$ or $L_i = \neg L_j$.*

Definition A.20. *A formula $\varphi \in F_P$ is in* Conjunctive Normal Form*, if it is a conjunction of disjunctions of literals.*

Consider a predicate CNF such that $\mathsf{CNF}(\varphi)$ holds if φ is in conjunctive normal form.

Lemma A.21. *A formula $\varphi \in F_P$ such that $\mathsf{CNF}(\varphi)$ is:*

- *valid, if all disjunctions are valid;*
- *contradictory, if some of the disjunctions are contradictory;*
- *satisfiable, otherwise.*

Any propositional formula is convertible in an equivalent formula in conjunctive normal form.

Theorem A.22. *For any formula $\varphi \in F_P$ there is a formula $\psi \in F_P$ such that $\varphi \equiv \psi$ and moreover, $\mathsf{CNF}(\psi)$.*

B Conversion to Horn Formula

Any basic Horn clause may be presented as an implication (*cf.* Lemma 2.2).

Lemma B.1. *Let L be a positive literal.*

1. $L \equiv \top \to L$
2. $\bigvee_{i=1}^{n} \neg L_i \equiv (\bigwedge_{i=1}^{n} L_i) \to \bot$
3. $\bigvee_{i=1}^{n} \neg L_i \vee L \equiv (\bigwedge_{i=1}^{n} L_i) \to L$

Proof. We use below standard equivalences of Propositional Logic. Recall that logical equivalence is a congruence relation.

1. $L \equiv \top \to L$

$$\begin{aligned} L &\equiv L \vee \bot \\ &\equiv L \vee \neg\neg\bot \\ &\equiv L \vee \neg\top \\ &\equiv \neg\top \vee L \\ &\equiv \top \to L \end{aligned}$$

2. $\bigvee_{i=1}^{n} \neg L_i \equiv (\bigwedge_{i=1}^{n} L_i) \to \bot$ The proof is by natural induction, using the following law.

$$(\varphi \to \gamma) \vee (\psi \to \gamma) \equiv (\varphi \wedge \psi) \to \gamma$$

Base case: n=1.

$$\bigvee_{i=1}^{n} \neg L_i = \neg L_1 \equiv \neg L_1 \vee \bot \equiv L_1 \rightarrow \bot$$

Inductive step:

$$\bigvee_{i=1}^{n+1} \neg L_i = \bigvee_{i=1}^{n} \neg L_i \vee \neg L_{n+1} \equiv ((\bigwedge_{i=1}^{n} L_i) \rightarrow \bot) \vee (L_{n+1} \rightarrow \bot) \equiv (\bigwedge_{i=1}^{n+1} L_i) \rightarrow \bot$$

The proof of the auxiliar law is easy.

$$\begin{aligned}(\varphi \wedge \psi) \rightarrow \gamma &\equiv \neg(\varphi \wedge \psi) \vee \gamma \\ &\equiv (\neg\varphi \vee \neg\psi) \vee \gamma \\ &\equiv (\neg\varphi \vee \neg\psi) \vee (\gamma \vee \gamma) \\ &\equiv (\neg\varphi \vee \gamma) \vee (\neg\psi \vee \gamma) \\ &\equiv (\varphi \rightarrow \gamma) \vee (\psi \rightarrow \gamma)\end{aligned}$$

3. $\bigvee_{i=1}^{n} \neg L_i \vee L \equiv (\bigwedge_{i=1}^{n} L_i) \rightarrow L$

   The proof is by natural induction.

   Base case: n=1.

$$\bigvee_{i=1}^{n} \neg L_i \vee L = \neg L_1 \vee L \equiv L_1 \rightarrow L$$

   Inductive step:

$$\begin{aligned}\bigvee_{i=1}^{n+1} \neg L_i \vee L &\equiv (\bigvee_{i=1}^{n} \neg L_i \vee \neg L_{n+1}) \vee (L \vee L) \\ &\equiv (\bigvee_{i=1}^{n} \neg L_i \vee L) \vee (\neg L_{n+1} \vee L) \\ &\equiv (\bigwedge_{i=1}^{n} L_i \rightarrow L) \vee (L_{n+1} \rightarrow L) \\ &\equiv (\bigwedge_{i=1}^{n+1} L_i) \rightarrow L\end{aligned}$$

$\square$

C Least Fixed-Point

We present here the proof of Lemma 3.1.

Lemma C.1. *Let φ be a Horn formula, i.e., $\varphi = \bigwedge_{i=1}^{n}(C_i \rightarrow L_i)$. The function $\mathscr{A}$ is:*

1. *increasing: $\mathscr{C} \subseteq \mathscr{A}(\varphi, \mathscr{C}) \subseteq \mathscr{C} \cup \bigcup_{i=1}^{n}\{L_i\}$;*
2. *and monotone: if $\mathscr{C} \subseteq \mathscr{D}$ then $\mathscr{A}(\varphi, \mathscr{C}) \subseteq \mathscr{A}(\varphi, \mathscr{D})$.*

Proof. The proofs of both cases are so similar that we present them together. If $\mathscr{A}(\varphi,\mathscr{C})=\mathscr{C}$, the results hold trivially. Otherwise, let $\mathscr{C}'=\mathscr{A}(\varphi,\mathscr{C})$ and $\mathscr{D}'=\mathscr{A}(\varphi,\mathscr{D})$. We proceed by natural induction on the number of clauses in φ.

Base case: let $\varphi = C \to L$. Since $\mathsf{set}(C) \subseteq \mathscr{C}$ (as $\mathscr{A}(\varphi,\mathscr{C}) \neq \mathscr{C}$), then $\mathscr{A}(C \to L,\mathscr{C}) = \mathscr{C}\cup\{L\}$. By hypothesis $\mathscr{C}\subseteq\mathscr{D}$, thus $\mathsf{set}(C)\subseteq\mathscr{D}$. Therefore, $\mathscr{C}\subseteq\mathscr{C}'=\mathscr{C}\cup\{L\}\subseteq\mathscr{D}\cup\{L\}=\mathscr{D}'$, and thus $\mathscr{A}$ is increasing and monotone.

Inductive step: let $\varphi = C \to L \wedge \bigwedge_{i=1}^{n+1}(C_i \to L_i)$, where $n \geq 0$. Assume, without loss of generality, that $\mathsf{set}(C)\subseteq\mathscr{C}$. Then,

$$\mathscr{C}' = \mathscr{A}(\varphi,\mathscr{C}) = \mathscr{A}(\bigwedge_{i=1}^{n+1}(C_i \to L_i),\mathscr{C}\cup\{L\})$$

If $\mathscr{A}(\bigwedge_{i=1}^{n+1}(C_i \to L_i),\mathscr{C}\cup\{L\}) = \mathscr{C}\cup\{L\}$, the results hold trivially. Otherwise, by induction hypothesis,

1. $\mathscr{C}\cup\{L\}\subseteq\mathscr{A}(\bigwedge_{i=1}^{n+1}(C_i \to L_i),\mathscr{C}\cup\{L\})\subseteq\mathscr{C}\cup\{L\}\cup\bigcup_{i=1}^{n}\{L_i\}$;
2. if $\mathscr{C}\cup\{L\}\subseteq\mathscr{D}\cup\{L\}$ then $\mathscr{A}(\bigwedge_{i=1}^{n+1}(C_i \to L_i),\mathscr{C}\cup\{L\})\subseteq\mathscr{A}(\varphi,\mathscr{D}\cup\{L\})$.

It is now simple to show the results. The function $\mathscr{A}$ is:

increasing - $\mathscr{C}\subseteq\mathscr{C}\cup\{L\}\subseteq\mathscr{C}'\subseteq\mathscr{C}\cup\{L\}\cup\bigcup_{i=1}^{n}\{L_i\}$; and

monotone - considering $\mathscr{C}\subseteq\mathscr{D}$, also $\mathscr{C}\cup\{L\}\subseteq\mathscr{D}\cup\{L\}$, and as $\mathscr{D}\subseteq\mathscr{D}\cup\{L\}$, we conclude $\mathscr{C}'\subseteq\mathscr{A}(\varphi,\mathscr{D})\subseteq\mathscr{A}(\varphi,\mathscr{D}\cup\{L\})$.

□

D Termination and complexity

Auxiliary notions. Henceforth, let $\varphi = \bigwedge_{i=1}^{n}(C_i \to L_i)$, where $n \geq 1$ be a Horn formula. Thus, each $\mathsf{set}(C_i)$ is a set of positive literals. Recall that a Horn formula may be regarded as a set of clauses.[3] Whenever $\varphi \in E_P$ is a Horn form such that $\varphi = \varphi_1 \wedge \varphi_2$, we may write $\varphi_1 \subseteq \varphi$. Then, for $\psi \subseteq \varphi$ and $\mathscr{C}\subseteq\mathscr{C}'$, when we write $\mathscr{A}(\varphi,\mathscr{C}) =^k \mathscr{A}(\psi,\mathscr{C}')$, the equality '$=^k$' denotes that the term on the right is obtained from the term on the left by executing k steps of the algorithm.

Main result. Theorem 3.4 is in fact a corollary of the following general result.

Theorem D.1. *For any Horn formula $\varphi = \bigwedge_{i=1}^{n}(C_i \to L_i)$ and any set $\mathscr{C}$ of literals such that $\{\top\}\subseteq\mathscr{C}\subseteq\{\top\}\cup\bigcup_{i=1}^{n}L_i$, there is a unique set of literals $\mathscr{C}'$ such that $\mathscr{A}(\varphi,\mathscr{C})=\mathscr{C}'$. Furthermore, the function $\mathscr{A}$ takes at most $n+1$ steps to yield $\mathscr{C}'$, where n is the number of clauses of φ.*

Proof. We proceed by natural induction on the number of clauses in φ.

Base case: since φ is a single clause; then, either $C = \top$ or $C = \bigwedge_{i=1}^{n} L_i$.

1. Case $\varphi = \top \to L$; therefore, as $\top \in \mathscr{C}$ by hypothesis, it is the case that

$$\mathscr{A}(\varphi,\mathscr{C}) = \mathscr{A}(\top,\mathscr{C}\cup\{L\}) = \mathscr{C}\cup\{L\}$$

[3] Any propositional formula in CNF determines univocally a set of sets of literals.

2. Case $\varphi = \bigwedge_{i=1}^{n} L_i \to L$; therefore, as $\mathscr{A}(\varphi,\mathscr{C}) = \mathscr{A}(\top,\mathscr{C}') = \mathscr{C}'$, where

$$\mathscr{C}' = \begin{cases} \mathscr{C} \cup \{L\}, & \text{if } \{L_i \mid \text{forall } 1 \le i \le n\} \subseteq \mathscr{C} \\ \mathscr{C}, & \text{otherwise} \end{cases}$$

In both cases the algorithm returns the result in two steps: one to analyse the clause and affect the resulting set; another to finish the execution, using the base case of the inductive definition. Notice that as $n = 1$, the execution of $\mathscr{A}$ takes exactly $n+1 = 2$ steps.

Inductive step: let $\varphi = \bigwedge_{i=1}^{n+1}(C_i \to L_i)$, where $n \ge 0$; notice that each $\{C_i\}$ is either $\{\top\}$ or a set of literals. Considering $\psi = \bigwedge_{i=1}^{n}(C_i \to L_i)$, then $\varphi = \psi \wedge (C_{n+1} \to L_{n+1})$. Assume, without loss of generality, that one chooses ψ such that

$$\mathscr{A}(\varphi,\mathscr{C}) =^k \mathscr{A}(\psi' \wedge (C_{n+1} \to L_{n+1}),\mathscr{C}') = \mathscr{C}''$$

where

1. $0 \le k \le n$;
2. $\psi' \subseteq \psi$, *i.e.*, it is a subset of clauses;
3. $C'' = \begin{cases} \mathscr{C}' \cup \{L_i\}, & \text{if } \{C_i\} \subseteq \mathscr{C}' \\ \mathscr{C}', & \text{otherwise} \end{cases}$

By induction hypothesis $\mathscr{C}'$ exists. Therefore, $\mathscr{C}''$ exists and is obtained from $\mathscr{C}'$ in two steps. Therefore, the execution of $\mathscr{A}$ takes $k+2$ steps and

$$k+2 \le n+2 = (n+1)+1.$$

□

E Unique least model

We present now the proof of Proposition 3.5. Let $\mathsf{SMB}(\varphi)$ denote the set of propositional symbols of the formula φ, inductively defined on the productions generating the Propositional Language (*cf.* Definition A.2). Notice first the following simple fact.

Lemma E.1. *For any Horn formula $\varphi \in E_P$, let $\mathscr{A}(\varphi,\{\top\}) = \mathscr{C}$. Then $\mathscr{C} \subseteq P \cup \{\bot,\top\}$.*

Proposition E.2. *For any Horn formula $\varphi \in E_P$, let $\mathscr{A}(\varphi,\{\top\}) = \mathscr{C}$ and $\bot \notin \mathscr{C}$. Then, $V \Vdash \varphi$ considering V such that $V(p) = 1$ for each $p \in \mathscr{C}$ and $V(q) = 0$ for each $q \in (P \setminus \mathscr{C})$.*

Proof. Let $\mathscr{C} = \{\top\} \cup \{p_i \mid 1 \le i \le n$, for some $n \ge 0\}$ (by the previous lemma). Using laws of Propositional Logic (in particular $(\varphi \to \psi) \wedge (\psi \to \gamma) \models \varphi \to \gamma$), one easily shows that $\{\varphi\} \models \bigwedge_{i=1}^{n}(\top \to p_i)$, and thus, if $V(p_i) = 1$ for each $1 \le i \le n$, by definition of satisfaction (*cf.* Definition A.7), it is the case that $V \Vdash \varphi$. □

F Correctness

F.1 Soundness

Lemma F.1. *Let $\varphi \in E_P$ be in Horn formula such that $\varphi = \psi \wedge (C \to \bot)$ and $\bot \in \mathscr{A}(\varphi,\{\top\})$. Then, $\{\varphi\} \models (\top \to \bot)$, being thus contradictory.*

Proof. Using laws of Propositional Logic (in particular $(\varphi \to \psi) \wedge (\psi \to \gamma) \models \varphi \to \gamma$, one easily shows the result. $\square$

Theorem F.2. *) For any Horn formula $\varphi \in E_P$:*

- $\perp \notin \mathscr{A}(\varphi, \{\top\})$*, only if φ it is satisfiable;*
- $\perp \in \mathscr{A}(\varphi, \{\top\})$*, only if φ it is contradictory.*

Proof. The first statement is a consequence of Proposition 3.5. We prove the second statement. Since by hypothesis, $\perp \in \mathscr{A}(\varphi, \{\top\})$, the contra-positive of Lemma 3.2 ensures that either $\varphi = \top \to \perp$ or there is a Horn formula γ such that $\varphi = \gamma \wedge (C_i \to \perp)$, for some $1 \leq i \leq n$. The case $\varphi = \top \to \perp$ yields immediatly the result, as $\varphi \equiv \perp$. Let us then consider the other case.

Let $\mathscr{A}(\varphi, \{\top\}) =^k \mathscr{A}(\psi, \mathscr{C})$ with:

1. $0 \leq k < n$;
2. $\top \in \mathscr{C}$ (by Lemma 1) and $\perp \notin \mathscr{C}$;
3. $\varphi = \varphi' \wedge \psi$ and either $\psi = \top \to \perp$ or $\psi = \psi' \wedge (C_i \to \perp)$, for some Horn form ψ'.

Assume, without loss of generality, that $\{C_i\} \subseteq \mathscr{C}$; then, by Lemma 1,

$$\mathscr{A}(\psi, \mathscr{C}) = \mathscr{A}(\psi', \mathscr{C} \cup \{\perp\}) \subseteq \mathscr{C} \cup \{\perp\}$$

Since $\varphi = \varphi' \wedge \psi' \wedge (C_i \to \perp)$, by Lemma F.1 we conclude that φ it is contradictory. $\square$

F.2 Completeness

Theorem F.3. *For any Horn formula $\varphi \in E_P$:*

- $\perp \notin \mathscr{A}(\varphi, \{\top\})$*, if φ it is satisfiable;*
- $\perp \in \mathscr{A}(\varphi, \{\top\})$*, if φ it is contradictory.*

Proof. The first statement is the contra-positive of the second statement of the previous theorem. The second is the contra-positive of the first statement of the previous theorem. $\square$