Exploiting Data Dependencies in Many-Valued Logics

Reiner Hähnle
University of Karlsruhe
Institute for Logic, Complexity and Deduction Systems
76128 Karlsruhe, Germany, reiner@ira.uka.de

ABSTRACT. The purpose of this paper is to make some practically relevant results in automated theorem proving available to many-valued logics with suitable modifications. We are working with a notion of many-valued first-order clauses which any finitely-valued logic formula can be translated into and that has been used several times in the literature, but in an ad hoc *way. We give a many-valued version of polarity which in turn leads to natural many-valued counterparts of Horn formulas, hyperresolution, and a Davis-Putnam procedure. We show that the many-valued generalizations share many of the desirable properties of the classical versions. Our results justify and generalize several earlier results on theorem proving in many-valued logics.*

KEYWORDS: many-valued logic, polarity, Horn formula, direct products of structures, resolution, Davis-Putnam procedure

Introduction

The purpose of this paper is to make some practically relevant results in automated theorem proving available to many-valued logics with suitable modifications. We are working with a notion of many-valued first-order clauses which we call *signed clauses* that has been used several times in the literature, but in a more *ad hoc* way. We will consider clauses of the form

$$\{S_1\, p_1, \ldots, S_m\, p_m\},$$

where the S_i are subsets of a finite or infinite set of truth values and the p_i are atomic formulas in the usual sense. In contrast to classical logic predicate symbols are interpreted as functions whose range is the set of truth values. The signed clause above is supposed to be satisfiable if there is an interpretation of the predicate and function symbols occurring in the atoms such that for all variable assignments at least one of the p_i has a truth value in S_i. There are no designated truth values, but it is shown in [H 94] that every satisfiability problem in any finitely-valued first-order logic with arbitrary designated truth values can be reduced to a satisfiability problem over signed clauses.

In this paper we give a many-valued version of polarity (for a definition and discussion of polarity in classical logic, see for example [M 82]) which in turn leads to natural many-valued counterparts of Horn formulas, hyperresolution, and a Davis-Putnam procedure. We show that the many-valued generalizations share many of the desirable properties of the classical versions. Our results justify and generalize several earlier results on theorem proving in many-valued logics.

We believe that the time has come for investigations such as the present one, because in the subfield of automated deduction in nonclassical logics affairs have by now reached a state, where it is understood quite well how suitable calculi for automated deduction systems can be constructed *in principle* for a wide selection of logics. There are resolution like calculi as well as tableaux like calculi for modal, many-valued, intuitionistic, non-monotonic and many other logics. The performance, however, of most systems based on those calculi is relatively poor when compared to state-of-the-art theorem provers for classical logic. One of the reasons, of course, is the inherently higher complexity of some nonclassical logics wrt their classical counterparts. Alas, this is not the whole story: let us have a look at some successful theorem proving paradigms from classical logic in order to see what might be missing.

- The *Davis-Putnam-Loveland (DPL) procedure* is among the fastest known methods for satisfiability checking on two-valued propositional clauses.
- Modern *Prolog* compilers achieve rates of hundreds of thousands logical inferences per second. Recall that the rule base of a standard Prolog program consists of first-order *definite Horn clauses*.
- In the resolution based theorem prover *Otter* the *hyperresolution* rule and *unit resulting* (UR) strategy often are keys to success.

Variable selection rules of the DPL procedure and hyperresolution rely explicitly on the notion that literals occurring in a clause can either be *positive* or *negative*. The same holds implicitly for Prolog programs which are based on Horn clauses.

These examples show that in classical logic formulas in clausal form induce *dependency relations* between the literals in a natural way which are exploited by many refinements of inference procedures. We need a clausal form that bears enough information for determining such data dependencies in many-valued logics as well, in other words, we need a notion of polarity of subformulas in many-valued logic.

In Section 1 of this paper we give formal definitions of some key notions of computational logic.

In Section 2 we formally define a many-valued version of polarity. In Sections 3, 4, and 5, respectively, we apply this definition in order to obtain in a natural way many-valued versions of Horn clauses, semantic clash resolution, and a DPL like procedure.

1 Definitions and Notation

In Definitions 1 to 5 we recapitulate some required concepts from classical logic.

Definition 1 *A* **first-order signature** Σ *is a triple* $\langle \mathbf{P}_\Sigma, \mathbf{F}_\Sigma, \alpha_\Sigma \rangle$*, where* $\mathbf{P}_\Sigma$ *is a non-empty family of* **predicate symbols**, $\mathbf{F}_\Sigma$ *is a possibly empty family of* **function symbols** *disjoint from* $\mathbf{P}_\Sigma$*, and* α_Σ *assigns a non-negative* **arity** *to each member of* $\mathbf{P}_\Sigma \cup \mathbf{F}_\Sigma$*. Let* $Term_\Sigma$ *be the set of* Σ**-terms** *over* **object variables** $Var = \{x_0, x_1, \ldots\}$*, and let* $Term^0_\Sigma$ *be the set of variable free terms called* **ground terms**.

Definition 2 *Let* Σ_0 *be a* **propositional signature**, *that is, a denumerable set of propositonal variables* $\{p_0, p_1, \ldots\}$*. In the propositional case we define the set of* **atoms** *to be* Σ_0*, in the first-order case let* $At_\Sigma = \{p(t_1, \ldots, t_n) |\ p \in \mathbf{P}_\Sigma, \alpha_\Sigma(p) = n, t_i \in Term_\Sigma\}$*. A* **literal** l *is either an atom* p *or a negated atom* $\neg p$*. In the first case* l *is assigned* **positive polarity**, *in the second case* **negative polarity**.

A **clause** *is a finite set of literals. A* **Horn clause** *is a clause that contains at most one literal with positive polarity.*

A formula in **clausal normal form (CNF)** *is a finite set of clauses which we require to have pairwise disjoint object variables in the first-order case. A* **Horn formula** *is a CNF formula that is made up from Horn clauses.*

By a 'Horn formula' and 'CNF formula' we mean always a universally closed formula in the first-order case as will become clear from the definitions.

Definition 3 *A* **propositional interpretation** I_0 *is a mapping from* Σ_0 *to* $\{0, 1\}$*. A* **first-order (Σ-)structure** $M_\Sigma = \langle D_M, I_M \rangle$ *consists of a non-empty domain* D_M *and an* **interpretation** I_M *that assigns to each function symbol* $f \in \mathbf{F}_\Sigma$ *a mapping* $I_M(f) : D_M^{\alpha_\Sigma(f)} \to D_M$ *and to each prediate symbol* $p \in \mathbf{P}_\Sigma$ *a relation in* $D_M^{\alpha_\Sigma(p)}$*. A* **variable assignment** *is a mapping* $\beta : Var \to D_M$.

In the following we omit the subscripts Σ, M, when no confusion can arise.

Definition 4 **Valuation functions** $val_{M,\beta}$ *for terms* $t \in Term_\Sigma$ *are defined inductively relative to a structure* M *and a variable assignment* β *as usual:*

$$val_{M,\beta}(x) = \beta(x) \text{ if } x \in Var$$

and

$$val_{M,\beta}(f(t_1, \ldots, t_m)) = I(f)(val_{M,\beta}(t_1), \ldots, val_{M,\beta}(t_m))$$

otherwise, provided $\alpha(f) = m$*.* $val_{M,\beta}$ *is extended to literals via:*

$$val_{M,\beta}(p(t_1, \ldots, t_m)) = 1 \text{ iff } \langle val_{M,\beta}(t_1), \ldots, val_{M,\beta}(t_m) \rangle \in I(p)$$

$$val_{M,\beta}(\neg p(t_1, \ldots, t_m)) = 1 \text{ iff } \langle val_{M,\beta}(t_1), \ldots, val_{M,\beta}(t_m) \rangle \notin I(p)$$

Definition 5 *A propositional literal* p *(*$\neg p$*) is* **satisfiable** *by an interpretation* I_0 *iff* $I_0(p) = 1$ *(*$I_0(p) = 0$*). A first-order literal* l *is satisfiable iff there is a* Σ*-structure* M *and a variable assignment* β *such that* $val_{M,\beta}(l) = 1$*.*

A propositional clause is satisfiable iff at least one literal is satisfiable. A first-order clause C *is satisfiable iff for some* Σ*-structure* M *for every variable assignment* β *at least one literal in* C *is satisfied. A (propositional) CNF formula* Φ *is satisfiable iff*

all clauses in Φ are simultaneously satisfiable by the same structure (interpretation) which is then called **model** *of Φ.*

Let Φ be a CNF formula and C a clause. Then C is a **logical consequence** *of Φ, in symbols $\Phi \models C$, iff every model of Φ is a model of C. Two sets of formulas are* **logically equivalent** *iff they have the same models.*

For many-valued semantics up to the atomic level only the following definitions must be changed:

Definition 6 *A* **truth value set** *N is either the finite set of equidistant rational numbers $\{0, \frac{1}{n-1}, \ldots, 1\}$ with cardinality n or the real unit interval. In the latter case we speak of ∞-valued logic. An* **(n-valued) propositional interpretation** *I_0 is a mapping from Σ_0 to N. An* **(n-valued) first-order (Σ-)structure** *$M_\Sigma = \langle D_M, I_M \rangle$ over a signature Σ is defined only for finite N and consists of a non-empty domain D_M and an interpretation I_M that assigns to each function symbol $f \in \mathbf{F}$ a mapping $I_M(f) : D_M^{\alpha(f)} \to D_M$, and to each prediate symbol $p \in \mathbf{P}$ a function $I_M(p) : D_M^{\alpha(p)} \to N$. In the many-valued case, $val_{M,\beta}$ is defined on atoms by $val_{M,\beta}(p(t_1, \ldots, t_m)) = I(p)(val_{M,\beta}(t_1), \ldots, val_{M,\beta}(t_m))$.*

2 Many-Valued Clauses and Polarity

Now we are going to define formally our notion of many-valued clause and many-valued CNF formula which we call *signed clause* and *signed formula*. We justify our particular choice as follows: first, it is demonstrated in [H 93a, H 94] that *any* satisfiability problem in finitely-valued first-order logic can be translated into a satisfiability preserving set of signed first-order clauses which, moreover, is polynomial in size to the input; second, [L-M-R 93, L-M-R 94] show that signed formulas can emulate annotated logics [L-H-S-dC 91] without significant overhead; third, [L-M-R 94] shows that inference on signed formulas can be used to model fuzzy inference; and forth, as can be observed below, all notions from classical logic we considered carry over naturally to the case of signed formulas.

In Lemma 22 below we show that in fact a sublanguage of the language of signed formulas called regular formulas is sufficient to express any signed formula. As a consequence, we will take regular formulas as the starting point of our investigation into computational properties of many-valued logic.

Definition 7 *Let $S \subseteq N$ and p be an atom. Then $S\ p$ is called a* **signed literal** *and S its* **sign**. *A finite set of signed literals is called a* **signed clause**. *A formula in signed CNF, or a* **signed formula** *for short, is a finite set of signed clauses which we require to have pairwise disjoint object variables in the first-order case.*

Definition 8 *A propositional signed literal $S\ p$ is* **satisfiable** *by an interpretation I_0 iff $I_0(p) \in S$. A first-order signed literal $S\ p$ is satisfiable iff there is a Σ-structure M and a variable assignment β such that $val_{M,\beta}(p) \in S$.*

A propositional signed clause is satisfiable iff at least one literal is satisfiable. A first-order signed clause C is satisfiable iff for some Σ-structure M for every variable assignment β at least one literal in C is satisfied.

A signed formula Φ is satisfiable iff all clauses in Φ are simultaneously satisfiable by the same (n-valued) interpretation, respectively, by the same (n-valued) structure which, in either case, is then called **(n-valued) model** *of Φ.*

Logical consequence *and* **logical equivalence** *are defined as in the classical case, but wrt many-valued models.*

Definition 9 *Let $\boxed{\geq i}$ denote the set $\{j \in N \mid j \geq i\}$ and let $\boxed{\leq i}$ denote the set $\{j \in N \mid j \leq i\}$. If a sign S is equal to either $\boxed{\geq i}$ or $\boxed{\leq i}$ for some $i \in N$, then it is called* **regular sign**.

Definition 10 *A* **regular clause** *is a signed clause made up of signed literals in which only regular signs occur. A literal $S\ p$ that occurs in a regular clause has* **positive polarity** *if $S = \boxed{\geq i}$ for some i, and it has* **negative polarity** *if $S = \boxed{\leq i}$ for some i.*

Usually we simply say that a literal $S\ p$ is positive (negative) when it has positive (negative) polarity.

3 Many-Valued Horn Clauses

Our aim is to define a natural many-valued generalization of Horn clauses. We claim that regular signed clauses are the proper basis for doing so. The definition is natural enough:

Definition 11 *A regular clause is a* **many-valued Horn clause** *if it contains at most one positive literal. A regular clause is said to be a* **positive (negative) regular clause** *if it contains only positive (negative) literals.*

Regular formulas and many-valued Horn formulas are defined in the obvious way like signed formulas above.

Example 12 *In the many-valued Horn formula Φ defined below all singleton clauses are positive while the fifth clause from the right is a negative clause.*

$$\begin{aligned}\Phi = \{&\{\boxed{\leq 0.3}\,p_1, \boxed{\leq 0.3}\,p_2, \boxed{\geq 0.7}\,p_3\},\ \{\boxed{\leq 0.6}\,p_3, \boxed{\leq 0.6}\,p_4, \boxed{\geq 0.4}\,p_5\},\\ &\{\boxed{\leq 0.4}\,p_5, \boxed{\leq 0.4}\,p_6, \boxed{\geq 0.6}\,p_8\},\ \{\boxed{\leq 0.5}\,p_5, \boxed{\geq 0.5}\,p_9\},\ \{\boxed{\leq 0.5}\,p_8, \boxed{\geq 0.5}\,p_{10}\},\\ &\{\boxed{\leq 0.4}\,p_{10}, \boxed{\geq 0.6}\,p_5\},\ \{\boxed{\leq 0.5}\,p_8, \boxed{\leq 0.5}\,p_9\},\ \{\boxed{\geq 0.6}\,p_1\},\\ &\{\boxed{\geq 0.6}\,p_2\},\ \{\boxed{\geq 0.8}\,p_4\},\ \{\boxed{\geq 0.4}\,p_6\}\}\end{aligned}$$

Remark 13 *1. Escalada Imaz & Manya Serres [EI-MS 94] give the following definition of a many-valued propositional Horn clause: if C is a classical Horn clause and $i \in \mathbb{R}$, then $\langle C; i\rangle$ is a many-valued Horn clause which is satisfiable iff $I(C) \geq i$ where $I(p \vee q) = \max\{I(p), I(q)\}$ and $I(\neg p) = 1 - I(p)$. Now consider a many-valued Horn clause in our sense of the form*

$$C_i = \{\boxed{\leq 1-i}\, p_1, \ldots, \boxed{\leq 1-i}\, p_k, \boxed{\geq i}\, p\}.$$

By definition, C_i is satisfiable by I iff one of its literals is satisfiable iff $1 - I(p_1) \geq i$ or ... or $1 - I(p_k) \geq i$ or $I(p) \geq i$, which is the case iff $\max\{1 - I(p_1), \ldots, 1 - I(p_k), I(p)\} \geq i$. The latter is equivalent to $I(\neg p_1 \vee \cdots \vee p_k \vee p) \geq i$, where $\vee, \neg$ are interpreted in the sense of [EI-MS 94], in other words iff $\langle \neg p_1 \vee \cdots \vee p_k \vee p; i \rangle$ is a satisfiable many-valued Horn clause in the sense of [EI-MS 94]. This shows that the many-valued Horn clauses of [EI-MS 94] are a special case of our notion. On the other hand, it is easy to see that the many-valued Horn formulas of [EI-MS 94] are less expressive than ours. Consider, for instance, the many-valued Horn formula $\{\boxed{\leq i}\, p, \boxed{\geq j}\, p\}$ for any $i < j$ such that $i \neq 1 - j$. Such a formula is clearly not equivalent to any set of Horn clauses in the sense of [EI-MS 94]: the set of many-valued functions that can be characterized with a conjunctive combination of clauses of the form $\{\boxed{\leq 1-j}\, p, \boxed{\geq j}\, p\}$, $\{\boxed{\geq k}\, p\}$, and $\{\boxed{\leq i}\, p\}$ does not include the function corresponding to the clause $\{\boxed{\leq i}\, p, \boxed{\geq j}\, p\}$ for $i < j$ and $i \neq 1 - j$.

2. *Schmitt [S 86] defines a first-order clause language based on literals that consist of atoms prefixed by arbitrary combinations of three-valued unary operators from the set $\{\neg, \sim\}$, where $\neg$ is defined as in [EI-MS 94] (cf. previous paragraph) and $\sim$ is defined as $val_{M,\beta}(\sim p) = \lceil 1 - val_{M,\beta}(p) \rceil$. Literals l in [S 86] are called satisfiable iff $val_{M,\beta}(l) \geq 1$. Satisfiability of clauses and formulas is as in our setting. In [S 86] literals of the form $p, \neg\sim p$ (and equivalent combinations) are called positive, literals of the form $\sim p, \neg p$ are called negative. Now, obviously, the (in Schmitt's sense) negative literal $\sim p$ is satisfiable with respect to $val_{M,\beta}$ in Schmitt's sense iff $val_{M,\beta}(\sim p) \geq 1$ iff $val_{M,\beta}(p) \in \{0, \frac{1}{2}\}$ iff $val_{M,\beta}(p) \leq \frac{1}{2}$ iff the (in our sense) negative literal $\boxed{\leq \frac{1}{2}}\, p$ is satisfiable in our sense. Similarly, all other combinations of $\neg, \sim$ can be characterized with regular signs and matching polarity. Hence, Schmitt's three-valued Horn clauses again are a special case of our notion.*

3.1 Computational Complexity

We begin our discussion of many-valued Horn formulas by defining a satisfiability checking procedure for them. It is a generalization of the algorithm given by Gallo & Urbani in [G-U 89] for classical propositional Horn formulas. Gallo & Urbani's algorithm in turn is a generalization of Dowling & Gallier's procedure [D-G 84], the latter being the base of the algorithms defined in [EI-MS 94].

Let Φ be a set of many-valued Horn clauses. We note that every Horn clause can be written in rule form, namely, if C is of the form

$$\{\boxed{\leq i_1}\, p_1, \ldots, \boxed{\leq i_k}\, p_k, \boxed{\geq i}\, p\},$$

then we rewrite it as

$$\boxed{>i_1}\, p_1 \wedge \cdots \wedge \boxed{>i_k}\, p_k \rightarrow \boxed{\geq i}\, p,$$

where $\boxed{>i}\, p$ abbreviates the negation of $\boxed{\leq i}\, p$.

Facts and negative clauses are adorned with pseudo-literals denoting truth and falsity, thus we write $T \rightarrow \boxed{\geq i}\, p$ for $\{\boxed{\geq i}\, p\}$ and $\boxed{>i_1}\, p_1 \wedge \cdots \wedge \boxed{>i_k}\, p_k \rightarrow F$ for $\{\boxed{\leq i_1}\, p_1, \ldots, \boxed{\leq i_k}\, p_k\}$, where T can be interpreted as $N\ p$ or $\boxed{\leq 1}\, p$ and F as $\emptyset\ p$ or $\boxed{>1}\, p$ (for arbitrary p) in the regular clause framework. We assign negative polarity to T and positive polarity to F. A literal of the form $\boxed{>i}\, p$ has negative polarity, because it occurs implicitly negated in the premise of an implication and expands to $\boxed{\leq i}\, p$.

Given a finite set $\mathbf{\Phi} = \{C_1, \ldots, C_m\}$ of such rules we associate a finite, labelled, directed graph $G_{\mathbf{\Phi}}$ with it whose node set consists of all the literals occurring in $\mathbf{\Phi}$ (including T and F[1]), and whose edges are defined as follows:

1. If l_i is a literal occurring in the body of a rule C_h in $\mathbf{\Phi}$, and l_j is its head literal, then there is an edge from l_i to l_j with label h in $G_{\mathbf{\Phi}}$.
2. For each pair of literals $\boxed{\geq i}\, p$ and $\boxed{>j}\, p$ such that $i > j$ there is an unlabelled edge from $\boxed{\geq i}\, p$ to $\boxed{>j}\, p$ in $G_{\mathbf{\Phi}}$.

Let us call edges of the former kind *inference edges* and edges of the latter kind *propagation edges* (in two-valued logic we have unsigned atoms in a rule base, so propagation edges are not needed). By definition, all edges going out from positive nodes are unlabelled propagation edges, all edges coming into positive nodes are labelled inference edges, and vice versa for negative nodes.

In Figure 1 the graph corresponding to the Horn clause set from Example 12 is shown. For sake of readability $\boxed{>i}\, p$ has been abbreviated with $> ip$ and $\boxed{\geq i}\, p$ with ip.

Let us denote with $\mathrm{lab}(l)$ the set of labels of edges entering into a positive node, with $\mathrm{prem}(h)$ the set of negative literals that occur in C_h, and, if D is a set of negative literals, with $\mathrm{prop}(D)$ the set positive literals that have an outgoing edge to some member of D. Now we are able to define a distance function d on the positive nodes of $G_{\mathbf{\Phi}}$ and T that indicates provability:

$$d(l_j) = 1 + \min_{h \in \mathrm{lab}(l_j)} \{\max\{d(l_i) |\ l_i \in \mathrm{prop}(\mathrm{prem}(h))\}\}$$

We adopt the convention that $\mathrm{prop}(\{T\}) = \{T\}$ and initially set $d(T) = 0$. If d is undefined for a positive node l, then we set $d(l) = \infty$. A positive node l is **reachable** iff $d(l) < \infty$. In Figure 1 the reachable nodes are framed.

Theorem 14 *Let l be a positive node which is reachable in $G_{\mathbf{\Phi}}$. Then $\mathbf{\Phi} \models l$.*

[1] We may assume that positive and negative clauses and, therefore, T and F occur in $\mathbf{\Phi}$, because otherwise $\mathbf{\Phi}$ is trivially satisfiable.

Proof: The proof is a straightforward induction over the distance of l similar as in [G-U 89]. ■

Corollary 15 *Φ is satisfiable iff F is unreachable in G_Φ.*

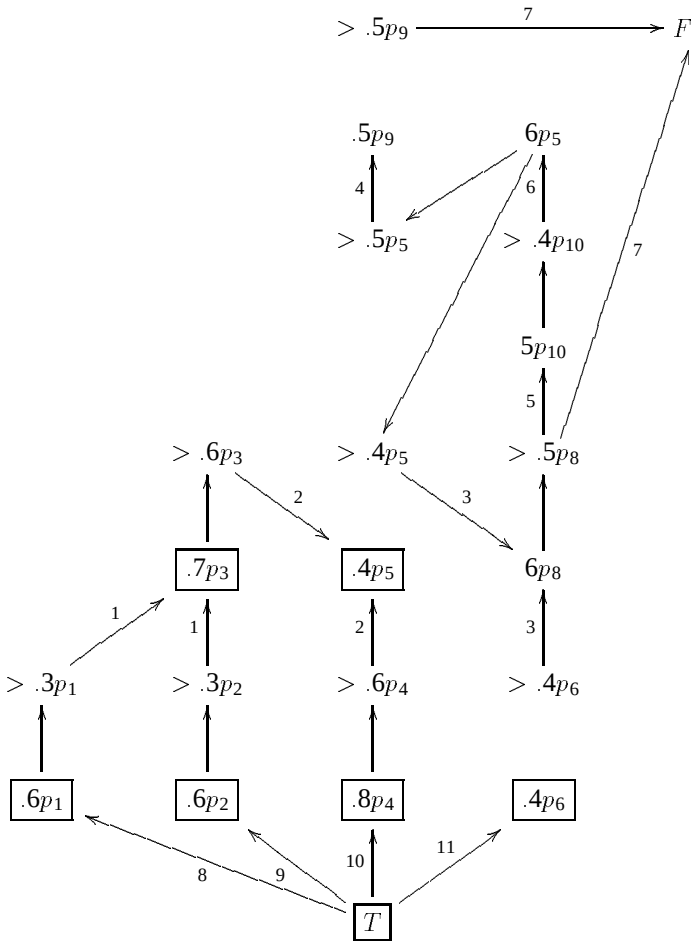

Figure 1: The graph corresponding to the Horn formula from Example 12. The framed nodes are contained in the reachability set.

Moreover, similarly as in [G-U 89], reachability can be checked in linear time wrt the size of G_Φ.

Unfortunately, the size of G_Φ from Φ is not linear wrt to the size of Φ as it is in classical logic. The culprit is easily identified as the second clause in the definition of

G_Φ which is not needed in the classical case. Obviously, up to mk edges are added by this clause, where m is the number of clauses in Φ, and k is the maximal number of clauses sharing a negative literal with the same atom and different annotation (if we assume without loss of generality that there is at most one negative literal with the same atom per clause).

It is possible, however, to represent the positive nodes with the same atom p as a list sorted according to increasing value of their annotations. The edges from the positive literals to a negative literal with atom p are not explicitly computed, instead the value of the highest reachable annotation in the list is stored and updated. Thus for each atom p only one edge is actually used at a time and the sorted list can be accessed in $\log k$ steps. Together we have a complexity $\mathcal{O}(|\Phi| + |\Sigma_\Phi| \log k)$ (where $|\Sigma_\Phi|$ is the number of different atoms occurring in Φ) which is the same as [EI-MS 94] gives for its more restricted class of many-valued Horn formulas.

When the number of truth values is finite (say, there are n truth values) k is bound by n. Hence the complexity again is linear. For an unbounded number of truth values we have an additional logarithmic factor in comparison to the classical case which is due to the fact that an atom can occur negatively with k different annotations.

As [EI-MS 94] remarks, this seems to be exactly the price we have to pay for the increased expressivity of real-valued Horn clauses. We have shown, however, that the notion of [EI-MS 94] is not quite as general as it could be.

3.2 Admitting Generic Examples

We now turn to the first of two justifications of our claim that many-valued Horn clauses based on regular clauses are a proper generalization of the classical case.

In [M 87] Makowsky gave an illuminating characterization of propositional and first-order Horn formulas based on the notion of generic examples. In the present subsection we demonstrate that at least for propositional logic this characterization can be extended to multiple truth values. First we introduce the classical notion:

Definition 16 *A classical propositional Σ_0-interpretation I is a* **generic model** *of a set of clauses Φ iff*

1. *I is a model of Φ and*
2. *For all $p \in \Sigma_0$: $I(p) = 1$ iff $\Phi \models p$ (that is, only such facts are in I which are contained in all models of Φ).*

Further, if $(I_i)_{i \in J}$ is a non-empty family of interpretations, then we define the interpretation $\bigcap_{i \in J} I_i$ via $\bigcap_{i \in J} I_i(p) = \min_{i \in J} I_i(p)$. With $I_{\min,\Phi}$ we abbreviate the interpretation $\bigcap\{I \mid I$ is a model of $\Phi\}$.

Definition 17 *A set of clauses* **admits generic models** *iff for every set of positive literals Δ the clause set $\Phi \cup \Delta$ either is not satisfiable or it has a generic model.*

Before we move on to the multi-valued case let us explain the meaning of these notions. Consider, for instance, $\Phi_1 = \{\{p, q\}\}, \Phi_2 = \{\{p\}, \{q\}\}, \Phi_3 = \{\{p, q, \neg r\}\}$.

Φ_1 does not have a generic model, because no fact is implied by Φ_1, but at least one of p, q must be true if Φ_1 is to be satisfied. Φ_2 and Φ_3 have generic models $\{p, q\}$ and $\emptyset$, respectively. Φ_2 does **admit** generic models, because its generic model is not affected by adding any positive literals. In contrast to this, Φ_3 does not admit generic models since adding the fact $\{r\}$ necessitates r to be true in any potential generic model for $\Phi_3 \cup \{\{r\}\}$, but neither p nor q does follow from $\Phi_3 \cup \{\{r\}\}$, although at least one of them must be true in a model.

Having a generic model is a robustness condition that corresponds to monotonicity of reasoning under the closed world assumption (CWA): when only Horn clauses are present, adding facts can only cause inference of new facts, whereas in the case of non-Horn clauses one might infer *disjunctive* information, for instance, Φ_1 can be inferred from $\Phi_3 \cup \{\{r\}\}$.

Thus, having generic models is a crucial property for formulas to have if one is interested in rule-based reasoning.

For the many-valued case we have to change the preceding definitions only slightly. Besides substituting 'many-valued' for 'classical' in Definition 16, all that has to be done is to replace clause 2 of Definition 16 with:

$$\text{For all } p \in \Sigma_0,\ i \in N:\ I(p) \geq i \text{ iff } \Phi \models \boxed{\geq i}\, p$$

To avoid technical complications with infinite sets of clauses we work with *finitely-valued* logics for the rest of this subsection.

The following results were stated and proved in [M 87] for classical logic. Here we give the many-valued generalizations:

Theorem 18 *Every set of regular clauses Φ has at most one generic model. If Φ has a generic model, then it is given by $I_{\min,\Phi}$. If Φ is Horn, then it is satisfiable iff it has a generic model.*

Proof: Without loss of generality assume that I_1, I_2 are generic for Φ, $I_1(p) = i_1$, $I_2(p) = i_2$ and $i_1 < i_2$. Thus $I_1(p) \not\geq i_2$, hence, since I_1 is generic, $\Phi \not\models \boxed{\geq i_2}\, p$. But from the genericity of I_2 we have that $\Phi \models \boxed{\geq i_2}\, p$, a contradiction. This proves that there is at most one generic model. The second claim is immediate from the definition. Now assume that Φ is Horn. The non-trivial direction is when Φ is satisfiable. It is easy to verify that the model given back by the satisfiability checking algorithm presented in the previous section is exactly $I_{\min,\Phi}$. ■

Theorem 19 *Let Φ be a set of regular clauses. Φ admits generic models iff Φ is logically equivalent to the set*

$$\Phi_H = \{C \mid C \text{ is a many-valued Horn clause and } \Phi \models C\}.$$

Proof: The non-trivial direction is $\Phi_H \models \Phi$. Assume D is a non-Horn clause that does not follow from Φ_H. Let $D = \{\boxed{\geq i_1}\, p_1, \ldots, \boxed{\geq i_k}\, p_k\} \cup E$, where E is a negative clause. Now, if we had $\Phi \models \boxed{\geq i_j}\, p_j$ for some j, then $\{\boxed{\geq i_j}\, p_j\} \in \Phi_H$, and then also $\Phi_H \models D$ which was assumed not to hold. Therefore, we may assume that

$$\Phi \models \boxed{\geq i_j}\, p_j \text{ for no } j \tag{1}$$

Since clauses that contain literals of the form $\boxed{\leq 1}\, p$ are equivalent to the Horn clause $\{T\}$, we may assume that D does not contain such literals. Therefore, the set

$$\Delta = \{\boxed{\geq j}\, p | \boxed{\leq i}\, p \in E, j > i \text{ arbitrary, but fixed}\}$$

is well-defined. Then $\Phi \cup \Delta$ has no generic model. To see this, assume there is a generic model I for $\Phi \cup \Delta$. By genericity of I, Δ, Φ, and in particular D are satisfied by I. By (1) $I(p_j) < i_j$ for all j, and from the satisfiability of Δ we have that $I(p) > i$ for all $\boxed{\leq i}\, p \in E$, thus D cannot be satisfied by I, a contradiction. ■

From Remark 13.1 it is obvious that the many-valued Horn formulas of [EI-MS 94] are not sufficient to characterize admission of generic models. Also there seems to be no natural variant of generic models under which it would be.

Makowsky [M 87] extended the notion of *having* vs *admitting* models to the first-order level. One of the results is that universal first-order Horn theories can be characterized by the fact that they admit initial term models. We expect that Makowsky's results can be carried over to the present case. Instead of pursuing this avenue further, however, we want to extend a more familiar characterization of universal first-order Horn clauses to the many-valued case in the following section.

3.3 Preservation Under Direct Products

One of the most important characterizations of Horn formulas[2] is the model theoretic one via direct products. One part of it is a generalization of the observation that many algebraic theories such as groups or rings are preserved under direct products which means for example that the direct product of groups again is a group. This preservation result can be shown in general for Horn formulas, thus any theory that can be characterized by Horn formulas is preserved under direct products. On the other hand, it turns out that if a formula is preserved under direct products then it must be equivalent to a Horn formula which gives a characterization of Horn formulas.

In the present section we show that this important characterization carries over to many-valued structures and Horn formulas. See for example [H 92, H 93b] for an extended discussion of direct products and related notions.

Definition 20 *Let $(M_i)_{i\in J} = \langle D_i, I_i\rangle$ be a non-empty family of many-valued structures over the same signature. Then we define the* **direct product** *$\prod_{i\in J} M_i = \langle D, I\rangle$ as follows: The domain D is the direct (Cartesian) product $D^J = \prod_{i\in J} D_i$, that is, the family of functions $g : J \to \bigcup_{i\in J} D_i$ such that $g(i) \in D_i$ for all $i \in J$. We define the interpretation of terms in $\prod_{i\in J} M_i$ via*

$$I(f)(t_1, \ldots, t_m) = \big(I_i(f)(t_1(i), \ldots, t_m(i))\big)_{i\in J} \text{ for } t_1, \ldots, t_m \in D$$

and the interpretation of atoms in $\prod_{i\in J} M_i$ via

[2] What we call Horn formulas is in model theory usually called more precisely *universal Horn sentence*.

$$I(p)(t_1,\ldots,t_m) = \min\{I_i(p)(t_1(i),\ldots,t_m(i))\mid i\in J\} \text{ for } t_1,\ldots,t_m\in D.$$

Recall that we work with only a finite number of truth values in the first-order case, therefore, it is sufficient to use min in the definition of $I(p)$.

Our definition of direct products of structures differs slightly from the one suggested in [W-W 69] which considers function-free languages and specifies only the value of the i-th component $I(p)(i)$ of $I(p)$, but leaves open the value of $I(p)$; on the other hand, a treatment of generalized quantifiers which we do not need here can be found there.

The definition of many-valued direct product coincides with the classical definition for the truth value set $N = \{0,1\}$ provided that two-valued predicates are considered as $\{0,1\}$-valued functions. The following is identical to the classical definition:

Definition 21 *A formula Φ is* **preserved under direct products** *iff for all non-empty families $(M_i)_{i\in J}$ of models for Φ, the direct product is also a model of Φ.*

Before we prove the main theorem of this section we state a lemma which ensures that it is sufficient to work with regular formulas rather than with signed formulas.

Lemma 22 *Every signed formula Φ is logically equivalent to some regular formula Φ_R.*

Proof: It is sufficient to replace each clause $C \in \Phi$ with a finite number of regular clauses $C_1,\ldots,C_m$ that are logically equivalent to C. Let $C = \{l_1,\ldots,l_k\}$ and assume that all literals up to l_r are regular, so $l_{r+1} = S\,p$ is the first non-regular literal in C. If S is an interval, that is, $S = \{\frac{a}{n-1},\ldots,\frac{b}{n-1}\}$ for certain $0 \le a \le b \le n-1$, then replace C with the two clauses $C' = \{l_1,\ldots,l_r,\boxed{\le\tfrac{b}{n-1}}\,p, l_{r+2},\ldots,l_k\}$ and $C'' = \{l_1,\ldots,l_r,\boxed{\ge\tfrac{a}{n-1}}\,p, l_{r+2},\ldots,l_k\}$. If S is not an interval, then take an arbitrary partition[3] $S_1 \cup \cdots \cup S_l$ of S into intervals S_i, and replace C with $C''' = \{l_1,\ldots,l_r,S_1\,p,\ldots,S_l\,p,l_{r+2},\ldots,l_k\}$, then apply the first kind of transformation to C'''. It is obvious that a finite number of such transformation steps must yield a set of regular clauses as desired.

It remains to show that both transformations do not change the models of a clause C. Obviously, it is enough to show that (i) $\models \{\frac{a}{n-1},\ldots,\frac{b}{n-1}\}\,p$ iff $\models \boxed{\ge\tfrac{a}{n-1}}\,p$ and $\models \boxed{\le\tfrac{b}{n-1}}\,p$ for all $0 \le a \le b \le n-1$; and (ii) $\models \{(S\cup S')\,p\}$ iff $\models \{S\,p, S'\,p\}$. Both, however, follow immediately from the definitions. ■

Theorem 23 *A signed formula Φ is preserved under direct products iff it is logically equivalent to some many-valued Horn formula Φ_H.*

[3] This is always possible since the number of truth values is finite.

Proof: Assume that $(M_i)_{i\in J}$ are models of a Horn formula Φ. Let M be the direct product of the M_i. We must show that M is a model of Φ. Let β be any variable assignment for M. By definition, each $C \in \Phi$ is satisfied for each variable assigment β_i in all M_i. First we assume that there is a negative literal $\boxed{\leq j}\, p \in C$ that is satisfied in some M_i (which is always the case for negative clauses). Hence, we have $\mathrm{val}_{M_i,\beta_i}(p) \leq j$, which, by definition of direct products, implies $\mathrm{val}_{M,\beta}(p) \leq j$, thus C is satisfied in M. If, on the other hand, C is positive, and for all i a positive literal in C is satisfied, then (since C is Horn) this positive literal must be the same literal for all i, say $\boxed{\geq j}\, p$. Again, by definition of direct products, $\mathrm{val}_{M_i,\beta_i}(p) \geq j$ for all i implies $\mathrm{val}_{M,\beta}(p) \geq j$, so M satisfies C.

For the other direction we adapt the proof of the classical result given in [C 81, Section VI.4]. First we reduce Φ: let $\mathcal{M}_\Phi$ be the class of models for Φ and let $\overline{\Phi}$ be the set of regular clauses modelled by $\mathcal{M}_\Phi$ (obviously, $\Phi \subseteq \overline{\Phi}$). For $C = \{l_1, \ldots, l_m\}$ denote with $C^{(k)}$ the clause $\{l_1, \ldots, l_{k-1}, l_{k+1}, \ldots, l_m\}$. Then, while there is a $C \in \Phi$ and k with $C^{(k)} \in \overline{\Phi}$, replace C with $C^{(k)}$. Call the resulting clause set Φ' the **reduction** of Φ. We may further assume that Φ' does not contain tautologous clauses of the form $\{\boxed{\geq 0}\, p, C\}$ or $\{\boxed{\leq 1}\, p, D\}$. It is easy to see that Φ' is still preserved under direct products: if $(M_i)_{i\in J}$ are models of $C' \in \Phi'$, then they are also models of a certain $C \in \Phi$ such that $C' \subseteq C$. Now, Φ is preserved under direct products, hence $\prod_{i\in J} M_i$ is a model of C. But $\prod_{i\in J} M_i \in \mathcal{M}_\Phi$ which was defined to model Φ', so we have that $\prod_{i\in J} M_i$ is a model of C' as well.

We claim that Φ' is many-valued Horn. Assume the contrary, which means that we have a clause $C = \{l_1, l_2, \ldots, l_m\} \in \Phi'$ where l_1 and l_2 are positive, say $l_1 = \boxed{\geq i_1}\, p_1$ and $l_2 = \boxed{\geq i_2}\, p_2$. Denote with M_k a model for C which is not a model for $C^{(k)}$. Since Φ' is reduced, such models must exist. By assumption, M_1 models $\{l_1, l_2, \ldots, l_m\}$ and there exists an assignment β_1 such that none of the literals $l_2, \ldots, l_m$ is satisfied by M_1 and β_1. By definition of satisfiability, we conclude that $\sim l_1 = \boxed{\leq i_1 - \frac{1}{n-1}}\, p_1$ is not satisfied by M_1 and β_1. Hence, none of $\sim l_1, l_2, \ldots, l_m$ is satisfied by M_1 and β_1. By similar reasoning, none of $l_1, \sim l_2, l_3, \ldots, l_m$ is satisfied by certain M_2 and β_2. By definition of direct products, we have that none of $l_1, l_2, l_3, \ldots, l_m$ is satisfied by $M = M_1 \otimes M_2$ and $\beta = (\beta_1, \beta_2)$, hence M is not a model of $C = \{l_1, l_2, \ldots, l_m\}$. But we did assume that both M_1 and M_2 are models of C, and by preservation of Φ' under direct products M must be a model of C as well which is a contradiction. ■

4 Refinements of Signed Resolution and their Completeness Proofs

In this section we return from Horn clauses to the full language of regular clauses in order to establish well known properties and techniques from the classical case for the latter. We are only concerned with propositional logic, because all deviations from classical logic occur on the ground level while lifting is done exactly as in the classical case and presents no new insights. This is due to the fact that our definition of signed clause is classical above the literal level and that we did not change the properties of terms and substitutions.

Consider the following resolution rule for general signed clauses:

$$\frac{\{S_1\ p\} \cup D_1 \quad \cdots \quad \{S_n\ p\} \cup D_n}{D_1 \cup \cdots \cup D_n} \quad \text{provided that } \bigcap_{i=1}^{n} S_i = \emptyset \tag{2}$$

It is shown to be refutationally complete for signed clauses together with the following merging rule in [H 94]:

$$\frac{\{S_1\ p, \ldots, S_m\ p\} \cup D}{\{(S_1 \cup \cdots \cup S_m)\ p\} \cup D} \tag{3}$$

The argument used there for establishing completeness was n-ary semantic trees. The same technique was used by Baaz & Fermüller [B-F 92] and Stachniak & O'Hearn [S-O 90] to prove related results.

In classical logic there are more ways to prove a resolution system complete some of which have advantages over semantic trees. Our aim is to transfer as many computational insights from classical logic to many-valued logic. Therefore, it is desirable to have a wide range of techniques for proving completeness. While semantic trees can be used to establish completeness for a number of many-valued resolution rules and refinements (for instance, ordered resolution, see [B-F 92]), they are either not suitable or more complicated than necessary for other refinements such as linear resolution restrictions.

In the following we propose a many-valued version of Anderson & Bledsoe's excess literal technique [A-B 70] which works particulary well with regular clauses and we show that it can be easily used to prove completeness of a version of semantic clash resolution for regular clauses.[4]

Definition 24 *Let Φ be an unsatisfiable set of signed clauses. We define the* **excess literal parameter** *k as $k(\Phi) = |\{l |\ l \in C \in \Phi\}| - |\{C |\ C \in \Phi\}|$.*

The idea of a completeness proof based on excess literals is to prove by induction on k that for any number of excess literals in an unsatisfiable clause set Φ the empty clause may be deduced from Φ.

Let us illustrate the principle with the following generalized version of negative hyperresolution [R 65] which acts on regular clauses:

Many-valued negative hyperresolution

$$\frac{\{\boxed{\leq i_1}\ p_1\} \cup D_1 \quad \cdots \quad \{\boxed{\leq i_n}\ p_n\} \cup D_n \qquad \{\boxed{\geq j_1}\ p_1, \ldots, \boxed{\geq j_n}\ p_n\} \cup E}{D_1 \cup \cdots \cup D_n \cup E} \tag{4}$$

provided $n \geq 1,\ i_l < j_l$ for all $1 \leq l \leq n,\ D_1, \ldots, D_n, E$ are negative

We note that every unsatisfiable set of regular clauses contains clauses as required for hyperresolution, because each such set must contain clauses with at least one

[4] The classical version of this result can be proved with a generalized version of semantic trees [K-H 69], but combining the latter with multiple truth values generates technical difficulties which are elegantly avoided in the excess literal technique.

positive literal and purely negative clauses. The first n input clauses are commonly called **electrons** and the non-negative input clause is called the **nucleus**.

Each hyperresolution step in turn produces a negative clause which can be subsequently used as an input for another hyperresolution step.

Theorem 25 *Let Φ be an unsatisfiable set of regular clauses. Then $\Box$ can be derived from Φ by a finite number of applications of many-valued negative hyperresolution.*

Proof: $k(\Phi) = 0$: That is, the number of literals in Φ is equal to the number of clauses in Φ. If $\Box \in \Phi$ we are done; otherwise, Φ must consist entirely of unit clauses. Now Φ is unsatisfiable, so there must be two clauses $\{\boxed{\leq i}\, p\}$ and $\{\boxed{\geq j}\, p\}$ such that $i < j$. Taking the second as the nucleus and the first as the single electron, hyperresolution immediately produces the empty clause.

$k(\Phi) > 0$: Again we assume that not already $\Box \in \Phi$. Moreover, there is a non-positive clause $C \in \Phi$ with at least two literals. For if not, then all negative literals in Φ appear in unit clauses. There must be a positive clause in Φ that produces immediately the empty clause with suitable negative unit clauses as electrons, otherwise, Φ would have a model that makes true all negative unit clauses and one literal not covered by any electron in each positive clause.

Hence we have a non-positive non-unit clause $C = \{\boxed{\leq i}\, p\} \cup D \in \Phi$ with $D \neq \Box$. Let $\Phi' = \Phi - \{C\}$, $\Phi_1 = \Phi' \cup \{D\}$, and $\Phi_2 = \Phi' \cup \{\{\boxed{\leq i}\, p\}\}$. By definition of clause satisfiability, both Φ_1 and Φ_2 are unsatisfiable. Moreover, $k(\Phi_1) < k(\Phi)$ and $k(\Phi_2) < k(\Phi)$, because the Φ_i were obtained by removing at least one literal from C. By the induction hypothesis we know that $\Box$ can be deduced from Φ_1 and from Φ_2.

Consider a hyperresolution proof of Φ_1. We note that if we replace each occurrence of D by $C = \{\boxed{\leq i}\, p\} \cup D$ in this proof then (i) still each step is a valid hyperresolution step and (ii) the last clause either is $\Box$ or $\{\{\boxed{\leq i}\, p\}\}$. In the first case we are done; the second case gives us a hyperresolution derivation of $\{\{\boxed{\leq i}\, p\}\}$ from Φ which can be extended to a proof of Φ using the induction hypothesis applied to Φ_2. ■

There are a few things worth pointing out in connection with this proof. First, one notices that our notion of many-valued clauses, polarity, and satisfiability were natural enough to take this proof virtually unaltered from [A-B 70]. Second, in [H 94] it is stated (without proof) that the rules

Regular resolution

$$\frac{\{\boxed{\geq i_1}\, p\} \cup D_1 \quad \cdots \quad \{\boxed{\geq i_n}\, p\} \cup D_n \qquad \{\boxed{\leq j}\, p\} \cup E}{D_1 \cup \cdots \cup D_n \cup E} \tag{5}$$

$$\text{provided that } j < \max\{i_k \mid 1 \leq k \leq n\}$$

Regular merging

$$\frac{\{\boxed{\leq i_1}\,p, \ldots, \boxed{\leq i_n}\,p,\ \boxed{\geq j_1}\,q, \ldots, \boxed{\geq j_m}\,q\} \cup D}{\{\boxed{\leq i}\,p,\ \boxed{\geq j}\,q\} \cup D} \qquad (6)$$

where $i = \max\{i_k \mid 1 \leq k \leq n\},\ j = \min\{j_l \mid 1 \leq l \leq m\}$

are complete on regular clauses. It is not unproblematic to prove this result with the semantic tree technique whereas the proof is quite straightforward with the excess literal technique and can be left to the reader in good conscience. It suffices to consider *any* clause with more than one literal in the induction step. Similarly, the excess literal technique provides an easy completeness proof for the signed resolution rule (2). It is interesting to note that from the excess literal proofs it is obvious that neither of the merging rules (3), (6) is needed for completeness. This is quite complicated to see in the proof using semantic trees and in fact has gone unnoticed in [H 94, L-M-R 93, M-R 93].

5 A Many-Valued Davis-Putnam Procedure

One of the most competitive decision procedures for satisfiability of classical clauses is the Davis-Putnam-Loveland procedure (DPL) [D-L-L 62] (for an assessment of its efficiency, see [B-B 92]). Below we give a formulation in pseudo Pascal code. The procedure call `dpl_sat(S)` terminates normally if `S` is unsatisfiable and it halts otherwise.

```
procedure dpl_sat(S: set of clause);
var L: literal;
begin
  S := unit_resolve(S);
  if S=∅ then
    halt;        /* S satisfiable */
  if □ ∉S then
    begin
      L := pick_literal(S);
      dpl_sat(S∪{L});
      dpl_sat(S∪{L̄})
    end
end;
```

```
function unit_resolve(S: set of clause): set of clause;
var L: literal;
var C: clause;
begin
  while ∃{L} ∈S do begin
    S := {C| L∉C∈S};    /* keep unsubsumed */
    S := {C−{L̄}| C∈S}   /* unit resolve */
  end
end;
```

With $\overline{\texttt{L}}$ we denote the complement of `L`. The function `pick_literal` (often called *branching rule* or *literal selection rule*) selects a literal that occurs in `S`. Its implementation turns out to be crucial for the performance of DPL. One of the best choices of `L` is the so-called *two-sided Jeroslow-Wang rule* [J-W 90, H 93c]. Given a set of clauses S we define for each literal L that occurs in S the following function:

$$J(L) = \sum_{L \in C \in S} 2^{-|C|}$$

Now define `pick_literal(S)` to be the literal that maximises $J(L) + J(\overline{L})$.

We close this paper by demonstrating that DPL including the two-sided Jeroslow-Wang branching rule can be extended to regular clauses naturally.

The most crucial difference that arises from the presence of more than two truth values is manifested in the branching rule. Of course, it would be possible to give a straightforward generalization that causes branching into n subproblems. This would be very inefficient. Therefore, the rationale in the design of a many-valued DPL for regular clauses should be (i) to keep the branching degree as low as possible and (ii), according to [H 93c], unit clauses generated by branching should produce as many new unit clauses as possible during unit resolution.

The proper choice for the many-valued version of $J(L)$, therefore, is to maximise $J(L) + J(\overline{L})$, where

$$J(L) = \sum_{\substack{\exists L' : L' \subseteq L \\ L' \in C \in S}} 2^{-|C|}$$

In the definition of J **subsumption of signed literals** is defined as follows: $S\ p \subseteq T\ q$ iff $p = q$ and $S \subseteq T$. The complement of a signed literal $S\ p$ is denoted with $\overline{S\ p}$ and defined to be $(N - S)\ p$.

Using this terminology in the code as well we can leave `dpl_sat` unaltered while `unit_resolve` must be changed only slightly:

```
function unit_resolve(S: set of clause): set of clause;
var L, L': literal;
var C: clause;
begin
   while ∃{L} ∈S do begin
      S := {C| ∄L'∈C∈S s.th. L⊆L'}; /* keep unsubs. */
      S := {C−{L'|L'∈C and L'⊆ L̄}|C∈S}  /* unit res. */
   end
end;
```

We remark that `unit_resolve` is complete for many-valued Horn clauses, but its run-time complexity is quadratic wrt the input in the worst case as opposed to the algorithm used in Section 3.1. We assume that using a similar representation of clauses as in Section 3.1 one could design a substitute for `unit_resolve` with linear worst-case complexity.

Example 26 *Consider the following set of clauses:*

$$\Phi = \{\{\boxed{\geq 1}p,\ \boxed{\leq 1}q\},\ \{\boxed{\geq 1}p,\ \boxed{\leq 0}p\},\ \{\boxed{\leq \tfrac{1}{2}}p,\ \boxed{\geq 1}q\},\{\boxed{\leq \tfrac{1}{2}}p,\ \boxed{\leq \tfrac{1}{2}}q\},$$
$$\{\boxed{\geq \tfrac{1}{2}}p,\ \boxed{\geq \tfrac{1}{2}}q\},\ \{\boxed{\leq \tfrac{1}{2}}p,\ \boxed{\geq \tfrac{1}{2}}q\},\ \{\boxed{\geq \tfrac{1}{2}}p,\ \boxed{\leq \tfrac{1}{2}}q\}\}$$

A many-valued DPL refutation of Φ *is depicted in Figure 2.* `pick_literal` *applied to* Φ *gives* $\boxed{\geq 1}p$ *which is shown in the root of the tree. There are no further branching steps necessary. In the linear subtrees the result of each round of simplification in* `unit_resolve` *is displayed.*

Conclusion

In this paper we tried to provide a conceptual and theoretical justification of a many-valued generalization of conjunctive clause formulas called regular formulas and a class of Horn formulas derived from them. The main advantage is that many important notions from computational logic in the classical case can be transferred extremely naturally due to a many-valued version of polarity which is built into regular formulas. This means that refinements of inference procedures for classical logic that exploit data dependencies can be generalized to many-valued logic.

Lemma 22 states that regular formulas are a sufficient generalisation of the classical case, because arbitrary signed formulas are not really necessary. Our many-valued Horn formulas have many desirable properties of classical Horn formulas, while a smaller class of Horn formulas with similar computational characteristics as defined in [EI-MS 94] is not sufficient. We are convinced that other properties of classical Horn formulas (see [H 92, EI-MS 94]) hold for our many-valued version as well. We believe that the present work at the same time unifies and justifies [S 86, L-M-R 93, M-R 93, EI-MS 94].

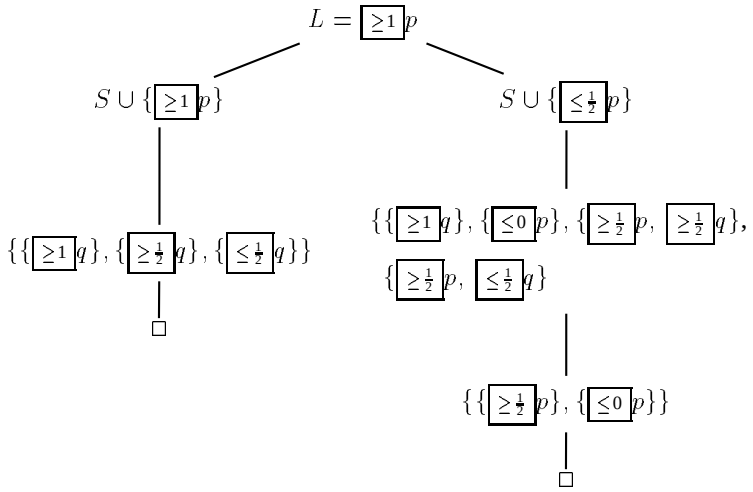

Figure 2: A many-valued DPL proof.

Acknowledgements

The detailed remarks of the referees led to a better presentation of some parts and to the removal of a notational error in the definition of direct products for many-valued structures.

References

[A-B 70] R. Anderson and W. Bledsoe. A linear format for resolution with merging and a new technique for establishing completeness. *JACM*, 17:525–534, July 1970. Reprinted in [S-W 83b].

[B-B 92] M. Buro and H. K. Büning. Report on a SAT competition. Reihe Informatik 110, FB 17—Mathematik/Informatik, Universität Paderborn, Nov 1992.

[B-F 92] M. Baaz and C. G. Fermüller. Resolution for many-valued logics. In A. Voronkov, editor, *Proc. Logic Programming and Automated Reasoning LPAR'92*, pages 107–118. Springer, LNAI 624, 1992.

[C 81] P. M. Cohn. *Universal Algebra*. Reidel, Dordrecht, second edition, 1981.

[D-G 84] W. Dowling and J. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulæ. *Journal of Logic Programming*, 3:267–284, 1984.

[D-L-L 62] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5:394–397, 1962.

[EI-MS 94] G. Escalada Imaz and F. Manyà Serres. The satisfiability problem for multiple-valued Horn formulæ. In *Proc. International Symposium on Multiple-Valued Logics, ISMVL'94, Boston/MA, USA*, pages 250–256. IEEE Press, Los Alamitos, 1994.

[G-U 89] G. Gallo and G. Urbani. Algorithms for testing the satisfiability of propositional formulae. *Journal of Logic Programmming*, 7(1):45–62, July 1989.

[H 92] W. Hodges. Logical features of Horn clauses. In D. M. Gabbay, C. J. Hogger, and J. A. Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 1: Logical Foundations, pages 449–503. Oxford University Press, 1992.

[H 93a] R. Hähnle. Short normal forms for arbitrary finitely-valued logics. In *Proceedings ISMIS'93, Trondheim, Norway*, pages 49–58. Springer LNCS 689, 1993.

[H 93b] W. Hodges. *Model Theory*, volume 42 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, 1993.

[H 93c] J. N. Hooker. Logic-based methods for optimization. A tutorial. Working Paper, GSIA, CMU Pittsburgh, December 1993.

[H 94] R. Hähnle. Short conjunctive normal forms in finitely-valued logics. *Journal of Logic and Computation, to appear*, 1994.

[J-W 90] R. G. Jeroslow and J. Wang. Solving propositional satisfiability problems. *Annals of Mathematics and Artificial Intelligence*, 1:167–187, 1990.

[K-H 69] R. Kowalski and P. J. Hayes. Semantic trees in automatic theorem proving. In *Machine Intelligence*, volume 4, pages 87–101. Edinburgh University Press, 1969. Reprinted in [S-W 83b].

[L-H-S-dC 91] J. J. Lu, L. J. Henschen, V. S. Subrahmanian, and N. C. A. da Costa. Reasoning in paraconsistent logics. In R. Boyer, editor, *Automated Reasoning: Essays in Honor of Woody Bledsoe*, pages 181–210. Kluwer, 1991.

[L-M-R 93] J. J. Lu, N. V. Murray, and E. Rosenthal. Signed formulas and annotated logics. In *Proc. International Symposium on Multiple-Valued Logics*, pages 48–53, 1993.

[L-M-R 94] J. J. Lu, N. V. Murray, and E. Rosenthal. Signed formulas and fuzzy operator logics. In *Proc. International Conference on Methodologies for Intelligent Systems, ISMIS'94, Charlotte/NC, USA*, pages 75–84. Springer, LNCS 869, 1994.

[M 82] N. V. Murray. Completely non-clausal theorem proving. *Artificial Intelligence*, 18:67–85, 1982.

[M 87] J. A. Makowsky. Why Horn formulas matter in computer science: Initial structures and generic examples. *Journal of Computer and System Sciences*, 34:266–292, 1987.

[M-R 93] N. V. Murray and E. Rosenthal. Signed formulas: A liftable meta logic for multiple-valued logics. In *Proceedings ISMIS'93, Trondheim, Norway*, pages 275–284. Springer LNCS 689, 1993.

[R 65] J. A. Robinson. Automatic deduction with hyper-resolution. *Int. Journal of Computer Math.*, 1:227–234, 1965. Reprinted in [S-W 83a].

[S 86] P. H. Schmitt. Computational aspects of three-valued logic. In J. H. Siekmann, editor, *Proc. 8th International Conference on Automated Deduction*, pages 190–198. Springer, LNCS, 1986.

[S-O 90] Z. Stachniak and P. O'Hearn. Resolution in the domain of strongly finite logics. *Fundamenta Informaticae*, XIII:333–351, 1990.

[S-W 83a] J. Siekmann and G. Wrightson, editors. *Automation of Reasoning: Classical Papers in Computational Logic 1957–1966*, volume 1. Springer-Verlag, 1983.

[S-W 83b] J. Siekmann and G. Wrightson, editors. *Automation of Reasoning: Classical Papers in Computational Logic 1967–1970*, volume 2. Springer-Verlag, 1983.

[W-W 69] J. Waszkiewicz and B. Węglorz. On products of structures for generalized logics. *Studia Logica*, XXV:7–13, 1969.