Nonmonotonic Reasoning with Quantified Boolean Constraints

Chris Pollett and Jeff Remmel

June 27, 1997

Overview

1. Motivation

2. Quantified Boolean formulas and the polynomial hierarchy

3. Our QBF_k formalisms

   (a) LP_k, CC_k, DL_k

4. Compactness of knowledge representation

   (a) Succinctness results

Motivation

Constraint Logic Programming (JL)

- Allow more general constraints to body of LP clauses. For instance, CLP($\mathbb{R}$) programs allow real inequalities and equalities as constraints.
- Constraints may be solved by special resources.
- Domain where constraints evaluated is fixed.

Constraint Programs (MNR)

- Is an extension of CLP paradigm.
- Domain is not fixed. An example use is in controlling a plant where set of applicable rules depends on plant's state at discrete intervals.
- Constraints need to be satisfied before evaluate remainder of clause.
- Constraints true in model consisting of atoms computed in process.
- Can still use special hardware.

Generalized constraints are a new source of complexity to be studied. We will discuss this complexity for the propositional case of a variety of nonmonotonic formalisms.

The Polynomial Hierarchy (PH)

$\mathbf{P} = \mathbf{\Delta_1^p} =$ deterministic p-time

$\mathbf{NP} = \mathbf{\Sigma_1^p}$ = nondeterministic p-time

$\mathbf{\Delta^p_{i+1} = P^{\Sigma^p_i}}$, $\mathbf{\Sigma^p_{i+1} = NP^{\Sigma^p_i}}$, $\mathbf{\Pi^p_i = co - \Sigma^p_i}$

$\mathbf{PH} = \cup \Sigma^p_k$

Open: PH$\not\downarrow$?

Quantified Boolean Formulas

$\mathbf{\Sigma_0^q = \Pi_0^q}$ - propositional formulas

$\mathbf{\Sigma^q_{k+1}} \supseteq \Pi^q_k$ closed under $(\exists x)$ where intended meaning of $(\exists x)A(x, \vec{b})$ is $A(0, \vec{b}) \vee A(1, \vec{b})$.

$\mathbf{\Pi^q_{k+1}} \supseteq \Sigma^q_k$ closed under $(\forall x)$ where intended meaning of $(\forall x)A(x, \vec{b})$ is $A(0, \vec{b}) \wedge A(1, \vec{b})$.

$\mathbf{QBF_k}$ - boolean combinations of Σ^q_k and Π^q_k.

$\mathbf{QBF_k(A)}$ - a QBF_k formula where allow atoms $x_1, \ldots, x_n \in A$. View $x_1, \ldots x_n$ as binary for a number and ask if in set A.

FACT: Validity of Σ^q_k-sentences is Σ^p_k-complete.

Our system for logic programming $\mathbf{LP_k}$

An $\mathbf{LP_k}$ program P is a finite list of clauses:

$$p \leftarrow a_1, \ldots, a_n : B_1(\vec{b}_1), \ldots, B_n(\vec{b}_m) \qquad (*)$$

where $p, a_1, \ldots, a_n$ are variables and $B \in QBF_k$. $\mathbf{LP_\infty} = \cup LP_k$. $\mathbf{LP_k(A)}$ - constraints from $QBF_k(A)$.

Stable Model Semantics Let $P \in LP_k$, M be a subset of P's vars. Let ν_M be truth assign. induced by M. Let P_M be obtained by deleting clauses whose constraints aren't satisfied by ν_M and by deleting the constraints from what's left. Let N_M be least model of P_M. M is a **stable model** of P if $M = N_M$.

Supported Model Semantics A **supported model** of $P \in LP_k$ is a truth assign. ν to vars in P such that $\nu(p) = 1$ iff $\exists$ a clause $(*)$ in P and $(\forall i, j)$ $\nu(a_i) = 1$, $\bar{\nu}(B(\vec{b_j})) = 1$. We write $\mathbf{LP_k^{sup}}$ if considering supported models. We write $\mathbf{LP_k^*}$ for programs with pairwise disjoint supported models.

Theorem

1. LP_0 is equivalent to logic progamming with negation.

2. Whether an LP_k program has a model is Σ^p_{k+1}-complete.

3. Whether an LP_∞ program has a model is $PSPACE$-complete.

Our system for circumscription $\mathbf{CC_k}$

Circumscribed models of a prop. formula are minimal models under inclusion. Could look at minimal models of QBF_k formulas. This doesn't separate constraints from computational component. Instead, $\mathbf{CC_k}$ program P is a finite list of clauses:

$$B(\vec{a}) \leftarrow: C(\vec{b})$$

with $B \in QBF_0$ and $C \in QBF_k$. $\mathbf{CC_\infty} = \cup CC_k$. $\mathbf{CC_k(A)}$ - constraints from $QBF_k(A)$

Semantics Let S be a subset of vars in P and let ν_S be corresponding var. assignment. Define $\bigwedge\!\!\!\bigwedge P_S := \bigwedge\!\!\!\bigwedge_{B \in P_S} B$ where P_S is

$$\{B \mid B \leftarrow: C \in P \wedge \bar{\nu}_S(C) = 1\}.$$

A **model** of P is a model of th 2nd-order formula $\bigwedge\!\!\!\bigwedge P_M \wedge \neg\exists m[\bigwedge\!\!\!\bigwedge P_m \wedge m \subset M]$.

Theorem

1. CC_0 is equivalent to prop. circumscription.
2. Whether a variable occurs in all models of a CC_k program is Π^p_{k+2}-complete.
3. The problem for CC_∞ is $PSPACE$-complete.

Our system for default logic $\mathbf{DL_k}$

A DL_k theory is a pair $\langle D, W \rangle$. Here D is a finite collection of default rules:

$$\frac{\alpha : B_1(\vec{b_1}), \ldots, B_m(\vec{b_m})}{\gamma}$$

where $\alpha, \gamma \in QBF_0$ and $B_i \in QBF_k$. W is a finite set of prop. formulas. $\mathbf{DL_\infty} = \cup DL_k$. $\mathbf{DL_k(A)}$ - constraints from $QBF_k(A)$

Stable Model Semantics A rule d is **S-applicable** if $B_i \cup S$ is consistent for each constraint in d. Form D_S by deleting non-S-applicable rule from D and deleting constraints from rest. Let $Cn^X(W)$ be all formulas provable from W using rules in X and prop logic. An **extension** for $\langle D, W \rangle$ is a set of formulas S such that $Cn^{D_S}(W) = S$. A **stable model** for $\langle D, W \rangle \in DL_k$ is a truth assign. satisfying an extension of $\langle D, W \rangle$.

<u>Supported Model Semantics</u> A rule d is **strongly S-applicable** if it is S-applicable and the prerequisite α of d is in S. Form $D_{S,w}$ as D_s but use strong S-applicability. A **weak extension** for $\langle D, W\rangle$ is a set of formulas S such that $Cn^{D_{S,w}}(W) = S$. A **supported model** for $\langle D, W\rangle \in DL_k$ is a truth assign. satisfying an weak extension of $\langle D, W\rangle$. We write DL_k^{sup} if considering stable models.

Theorem

1. DL_0 is the same as usual default logic.
2. Whether $\langle D, W\rangle \in DL_k$ has an extension is Σ_{k+2}^p-complete.
3. The problem for DL_∞ it is $PSPACE$-complete.

Compactness of knowledge representation

Definition (GKPS, CDS) Let A and B be reasoning formalisms. Then A *as succinct as* B, written $B \leq_s A$ if: For each ϕ_B in B there is a knowledge base ϕ_A in A such that
(a) ϕ_B and ϕ_A use *free* variables and have the same models
(b) the size of ϕ_A is polynomial in the size of ϕ_B.
We write $A \not\leq_s B$ if (a) and (b) fail to hold.

Definition We say A *is as weak succinct as* B, written $B \leq_{ws} A$ if the conditions above hold but condition (a) is replaced with
(a') ϕ_A contains all of ϕ_B's variables and all models of ϕ_A are expansions of models of ϕ_B.

• GKPS give a succinctness hierarchy among prop logic, Horn logic, circumscription and default logic which is strict provided PH$\not\downarrow$.

Hierarchies of Knowledge Formalisms

(a) $LP_k <_s DL_k$

(b) $LP_k^* <_s CC_k <_s DL_k$

(c) $LP_k^* \equiv_{ws} LP_k^{sup} \equiv_{ws} LP_k <_{ws} CC_k \leq_{ws}$
$\leq_{ws} DL_k^{sup} \equiv_{ws} DL_k \equiv_{ws} LP_{k+1}.$

- Strictness in above under assumption PH$\not\downarrow$.

(d) $\exists A$, $LP_k(A) <_{ws} CC_k(A) <_s DL_k(A)$
(e) $LP_\infty \equiv_{ws} CC_\infty \equiv_{ws} DL_\infty$

Theorem If K is a reasoning formalism with Δ_{k+1}^p-model checking then $CC_k \not\leq_s K$ unless $\Sigma_{k+1}^p \subseteq \Delta_{k+1}^p/poly$.

Theorem Suppose K is a reasoning formalism for which $Model_{\phi_K}(\vec{x})$ can be expressed as a QBF_k formula. Then $K \leq_{ws} LP_k$.