# Logic-Based Bidding Languages with Intermediate Complexity

**Andreas Witzel**
ILLC, University of Amsterdam
Plantage Muidergracht 24
1018TV Amsterdam, Netherlands
and CWI, Kruislaan 413,
1098SJ Amsterdam, Netherlands

**Abstract**

Logic-based bidding languages are used for preference representation in combinatorial auctions. Given a set of propositional formulas with associated weights, finding a valuation that maximizes the sum of the weights which are associated to satisfied formulas is a canonical problem in this context. The general case is intractable, and natural restrictions of the languages tend to either leave the complexity unchanged or reduce it to triviality. After proposing a different decision problem than the one considered in existing research, we use a new approach to find $P$-complete languages.

## Goal Base Decision Problem

### The Function Problem

In combinatorial auctions using a class $C$ of weighted propositional formulas $(\varphi, w)$ as possible bids, the winner determination problem is as follows: Given a set $S \in C$, find an assignment $v_{max}$ for the propositional letters such that $v_{max}(S)$ is maximal, where

$$v(S) := \sum_{(\varphi,w)\in S \text{ with } v(\varphi)=\top} w.$$

### The Decision Problem used so far

The decision problem formulation used so far was: Given a set $S \in C$ and a number $k$, is $v_{max}(S) \geq k$?

We argue that this decision problem does not properly represent the above function problem. For example, consider the class

$$C = \{S | S \text{ is a satisfiable set of formulas with weights } \geq 0\}$$

Then obviously the decision problem is trivial (take the sum of all weights and check whether it is $\geq k$), while the function problem is possibly not trivial (prove this?), and it is not possible to construct a solution to the function problem by solving the decision problem.

### Proposing a Different Decision Problem

A function problem and an associated decision problem should be related in the sense that solving one enables one to solve the other (see and maybe some better references...). This does not hold for the decision problem used so far.

We therefore propose the following decision problem: Given a set $S$ and a propositional letter $p$, is $v_{max}(p) = \top$ under the least (according to some linear order over the maximizing assignments) maximizing assignment $v_{max}$? (counter-example for a definition requiring just some maximizing assignment: $\{(1, p \wedge \neg q), (1, \neg p \wedge q)\}$)

A solution procedure for this decision problem can be called $n$ times (where $n$ is the number of available propositional variables) to construct a solution to the original function problem.

## Propositional Logic Programming

**Definition 1** *By $PS$ we denote the set of all propositional symbols (atoms) of the language under consideration.*

**Definition 2** *For a set of atoms $A$ and a propositional formula $\varphi$, we write $A \models \varphi$ iff the valuation assigning $\top$ to all elements of $A$ and $\bot$ to all other atoms satisfies $\varphi$. For a set $S$ of propositional formulas, we write $A \models S$ iff $A \models \varphi$ for all $\varphi \in S$. We use the set and the valuation notions interchangeably.*

**Definition 3** *A* (strict/general) Horn clause *is a disjunction containing (exactly/at most) one positive literal.*

**Definition 4** *For a set $S$ of strict Horn clauses, the* least model $LM(S)$ *of $S$ is the (unique) smallest set $A$ such that $A \models S$.*

**Fact 5** *Existence of a unique least model is a result from logic programming.*

**Definition 6 (Propositional Logic Programming)**

**Given:** *A set $S$ of strict Horn clauses and $p \in PS$*

**Question:** *Is $p \in LM(S)$?*

**Fact 7 (from (Dantsin *et al.* 2001, p. 385))** *Propositional Logic Programming is $P$-complete.*

## Propositional Logic Programming Goal Bases

**Definition 8** *The class $\mathcal{G}_{PLP}$ of* Propositional Logic Programming Goal Bases *consists of all goal bases*

$$G = \bigcup_{i=1}^{n} \{(\varphi_i, w_i)\} \cup \bigcup_{p\in PS} \{(p, -\tfrac{m}{|PS|+1} b)\} \ ,$$

*where $w_i > 0$ for all $i$, $m = \min_{i=1}^{n}\{w_i\}$, and $LP(G) := \{\varphi_1, \dots \varphi_n\}$ is the* underlying logic program *consisting of all positively weighted formulas.*

**Fact 9** *The negative weights associated to the single atoms sum up to an absolute value less than any $w_i$. I.e., for any $G \in \mathcal{G}_{PLP}$ and all $i$, $w_i > \sum_{p \in PS} \frac{m}{|PS|+1}$.*

**Corollary 10** *The (unique) maximizing valuation of any $G \in \mathcal{G}_{PLP}$ is the least model of the underlying logic program, i.e. $LM(LP(G))$.*

*Proof.* $v := LM(LP(G))$ obviously satisfies all formulas of $G$ that have positive weights. Due to $v$ being a *least* model and Fact 9, no subset of $v$ gets a higher value; due to the negative weights associated to single atoms, no superset of $v$ gets a higher value; and due to Fact 5, $v$ is unique. □

**Fact 11** *The decision problem from* **??** *for PLP Goal Bases is in $P$.*

*Proof.* Given $G \in \mathcal{G}_{PLP}$ and $p \in PS$, $LP(G)$ can be computed in linear time, and then $p \in LM(LP(G))$ is decidable in polynomial time due to Fact 7. According to Corollary 10, this yields the answer to the original problem. □

**Fact 12** *Propositional Logic Programming can be reduced in logarithmic space to the decision problem from* **??** *for PLP Goal Bases.*

*Proof.* Given a logic program $S = \{\varphi_1, \dots, \varphi_n\}$ and $p \in PS$, define

$$G := \bigcup_{i=1}^{n} \{(\varphi_i, 1)\} \cup \bigcup_{p \in PS} \{(p, -\tfrac{1}{|PS|+1})\} \ .$$

Obviously, $G \in \mathcal{G}_{PLP}$, and due to Corollary 10, the solution to the PLP Goal Base decision problem instance $(G, p)$ is also the solution to the Propositional Logic Programming decision problem instance $(S, p)$. □

**Corollary 13** *The decision problem from* **??** *for PLP Goal Bases is $P$-complete.*

**Note 14** *Maybe there are be more natural ways to force the maximizing valuation to correspond to the* least *model.*

*Couldn't this even be a desideratum for the allocation mechanism in order to make sure not to give out things unnecessarily to someone who doesn't profit from them, while maybe someone else would have?*

## HORNSAT Approach

Maybe the following is slightly more natural...

**Fact 15** *Deciding satisfiability of a set of general Horn clauses is $P$-complete (Greenlaw, Hoover, & Ruzzo 1992).*

**Definition 16** *The class $\mathcal{G}_{HS}$ of* HORNSAT Goal Bases *consists of all sets $G$ of weighted general Horn clauses with positive weights, subject to the following (unintuitive) condition: Let $w_i$ denote the weights of the strict Horn clauses in $G$ and $w'_j$ denote the remaining weights. Then we require that $\sum_j w'_j < \min_i\{w_i\}$. That is, the sum of weights of non-strict clauses (i.e. those containing no positive literal) is less than the least weight associated to some strict clause.*

**Fact 17** *The decision problem from* **??** *for HS Goal Bases is in $P$.*

*Proof.* Given $G \in \mathcal{G}_{HS}$, use e.g. unit propagation to find satisfying assignment if it exists. If it does exist, this is the maximizing assignment since all weights are positive. If it does not exist, let $G' \subset G$ be the subset of all strict Horn clauses. Due to the unintuitive condition from Definition 16, $LM(G')$ is a maximizing assignment for $G$, since it satisfies all strict Horn clauses, and satisfies the most non-strict Horn clauses among all assignments which satisfy all strict Horn clauses.

Any other maximizing assignments are supersets of $LM(G')$, so the $\subsetneq$ relation is a linear order over maximizing assignments with $LM(G')$ as least element, and thus we can use this assignment to solve the decision problem from **??** for HS Goal Bases. □

**Fact 18** *HORNSAT can be reduced in logarithmic space to the decision problem from* **??** *for HS Goal Bases.*

*Proof.* Given a set $S = \{\varphi_1, \dots, \varphi_n, \varphi'_1, \dots, \varphi'_m\}$ of strict ($\varphi_i$) and non-strict ($\varphi'_i$) Horn clauses, build the HS Goal Base

$$G := \bigcup_{i=1}^{n} \{(\varphi_i, 1)\} \cup \bigcup_{i=1}^{m} \{(\varphi'_i, \tfrac{1}{m+1})\} \ ,$$

obtain the maximizing assignment by solving the decision problem for $G$ and each $p \in PS$, and check whether it satisfies all formulas in $G$. Since the assignment is maximizing and all weights are positive, it will do so iff $G$ is satisfiable. □

**Corollary 19** *The decision problem from* **??** *for HS Goal Bases is $P$-complete.*

**Note 20** *The unintuitive condition in Definition 16 is only used for Fact 17. Maybe it is possible to come up with a more intuitive condition ensuring that we stay in $P$.*

## References

Dantsin, E.; Eiter, T.; Gottlob, G.; and Voronkov, A. 2001. Complexity and expressive power of logic programming. *ACM Computing Surveys* 33:374–425.

Greenlaw, R.; Hoover, H. J.; and Ruzzo, W. L. 1992. A compendium of problems complete for P.