Qualitative Decision Making and Answer Set Programming Extended Abstract

Gerhard Brewka
Universität Leipzig, Institut für Informatik
Augustusplatz 10-11, 04109 Leipzig, Germany
brewka@informatik.uni-leipzig.de

1 Introduction

In a recent paper [BBL02] a propositional logic called Qualitative Choice Logic (QCL) was introduced. The logic contains a new connective $\times$ representing ordered disjunction. Intuitively, $A \times B$ stands for: if possible A, but if A is impossible then (at least) B. This connective allows context dependent preferences to be represented in a simple and elegant fashion. In this paper we show how to combine ideas underlying QCL with logic programming and how to use the resulting framework for qualitative decision making. The semantical framework in which the investigation will be carried out is that of answer set semantics [GL91]. Intuitively, answer sets are sets of literals describing plausible states of the world given the knowledge represented in a program. There are several interesting AI applications of answer set programming, for instance in planning and configuration. One of the reasons for this success is the availability of highly efficient systems for computing answer sets like *smodels* [NS97] and *dlv* [ELM$^+$98].

The basic intuition underlying our approach can be described as follows: we will use the ordered disjunctions in rule heads to induce a preference ordering on the answer sets of a program. Under certain conditions reasoning from most preferred answer sets yields optimal problem solutions. In more general decision making settings the preference relation on answer sets provides a basis for best possible choices given a specific decision strategy.

2 Logic programs with ordered disjunction

Logic programming with ordered disjunction is an extension of logic programming with two kinds of negation (default and strong negation) [GL91]. The new connective $\times$ representing ordered disjunction is allowed to appear in the head of rules only. A (propositional) $LPOD$ consists of rules of the form

$$C_1 \times \ldots \times C_n \leftarrow A_1, \ldots, A_m, \mathrm{not}\, B_1, \ldots, \mathrm{not}\, B_k$$

where the C_i, A_j and B_l are ground literals.

The intuitive reading of the rule head is: if possible C_1, if C_1 is not possible then C_2, ..., if all of $C_1, \ldots, C_{n-1}$ are not possible then C_n. The literals C_i are called choices of the rule. Extended logic programs with two negations are a special case where $n = 1$ for all rules. As usual we omit $\leftarrow$ whenever $m = 0$ and $k = 0$, that is, if the rule is a fact. Moreover, rules of the form $\leftarrow body$ (constraints) are used as abbreviations for $p \leftarrow body, \mathrm{not}\, p$ for some p not appearing in the rest of the program. The effect is that no answer sets containing $body$ exist.

Since ordered disjunction is a particular prioritized form of disjunction it seems like a natural idea to base the semantics of $LPOD$s on one of the standard semantics for disjunctive logic programs, for instance Gelfond and Lifschitz's semantics [GL91].

Unfortunately, this doesn't work. The problem is that most of these semantics have minimality built in. For instance, according to Gelfond and Lifschitz, S is an answer set of a disjunctive logic program P iff S is a minimal set of literals which is logically closed, and closed under the S-reduct of P. The S-reduct of P is obtained from P by (1) deleting all rules r from P such that $\mathrm{not}\, B_j$ in the body of r and $B_j \in S$, and (2) deleting all default negated literals from the remaining rules. A set of literals S is closed under a rule r if one of the literals in the head of r is in S whenever the body is true in S (see [GL91] for the details).

In this approach answer sets are minimal: if S_1 and S_2 are answer sets of a disjunctive program P and $S_1 \subseteq S_2$, then $S_2 \subseteq S_1$. Minimality is not always wanted for $LPOD$s. Consider the following two facts:

1) $A \times B \times C$
2) $B \times D$

The single best way of satisfying both ordered disjunctions is obviously to make A and B true, that is, we would expect $\{A, B\}$ to be the single preferred answer set of this simple $LPOD$. We thus have to use a semantics which is not minimal.

Our semantics is a modification of a semantics proposed by Sakama and Inoue [SI94].

Definition 1 *Let $r = C_1 \times \ldots \times C_n \leftarrow body$ be a rule. For $k \leq n$ we define the kth option of r as*

$$r^k = C_k \leftarrow body, \mathrm{not}\, C_1, \ldots, \mathrm{not}\, C_{k-1}.$$

Definition 2 *Let P be an $LPOD$. P' is a split program of P if it is obtained from P by replacing each rule in P by one of its options.*

Split programs do not contain ordered disjunction. We thus can define:

Definition 3 *Let P be an $LPOD$. A set of literals A is an answer set of P if it is a consistent answer set of a split program P' of P.*

To distinguish between more and less intended answer sets we introduce the degree of satisfaction of a rule in an answer set:

Definition 4 *Let S be an answer set of an $LPOD$ P. S satisfies the rule*

$$C_1 \times \ldots \times C_n \leftarrow A_1, \ldots, A_m, \text{not } B_1, \ldots, \text{not } B_k$$

- *to degree 1 if $A_j \notin S$, for some j, or $B_i \in S$, for some i,*
- *to degree j $(1 \leq j \leq n)$ if all $A_j \in S$, no $B_i \in S$, and $j = min\{r \mid C_r \in S\}$.*

We use the degrees of satisfaction of a rule to define a preference relation on answer sets. There are different ways of doing this. In [BBL02] a lexicographic ordering of models based on the number of premises satisfied to a particular degree was proposed. This lexicographic ordering has a highly syntactic flavour. Therefore, we will use here a somewhat more cautious preference relation (in the sense that fewer answer sets are considered better than others) based on set inclusion of the rules satisfied to certain degrees:

Definition 5 *For a set of literals S, let $S^i(P)$ denote the set of rules in P satisfied by S to degree i. Let S_1 and S_2 be answer sets of an $LPOD$ P. S_1 is preferred to S_2 ($S_1 > S_2$) iff there is i such that $S_2^i(P) \subset S_1^i(P)$, and for all $j < i$, $S_1^j(P) = S_2^j(P)$.*

3 Decision Making using LPODs

In this section we describe a methodology for qualitative decision making based on $LPOD$s. The basic idea is to use $LPOD$s to describe possible actions or decisions and their consequences, states of the world and desired outcomes. The representation of desires induces, through ordered disjunction, a preference ordering on answer sets representing their desirability. Based on this preference ordering an ordering on possible decisions can be defined based on some decision strategy.

We will use Savage's famous rotten egg example [Sav54] to illustrate this methodology. An agent is preparing an omelette. 5 fresh eggs are already in the omelette. There is one more egg. It is uncertain whether this egg is fresh or rotten. The agent can

- add it to the omelette which means the whole omelette may be wasted, or
- throw it away, which means one egg may be wasted, or
- put it in a cup, check whether it is ok or not and put it to the omelette in the former case, throw it away in the latter. In any case, a cup has to be washed if this option is chosen.

In this example, the set of literals C representing possible decisions is the set of literals built from $\{in{-}omelette, in{-}cup, throw{-}away\}$. Here are the rules which generate the possible decisions and states of the world:

$in{-}omelette \leftarrow \text{not } in{-}cup, \text{not } throw{-}away$
$in{-}cup \leftarrow \text{not } in{-}omelette, \text{not } throw{-}away$
$throw{-}away \leftarrow \text{not } in{-}cup, \text{not } in{-}omelette$
$rotten \leftarrow \text{not } fresh$
$fresh \leftarrow \text{not } rotten$

It is not necessary to specify that the different actions and states of the egg are mutually exclusive. It is guaranteed by the rules that only one of the exclusive options is contained in an answer set.

We next define the effects of the different choices:

$5{-}omelette \leftarrow throw{-}away$
$6{-}omelette \leftarrow fresh, in{-}omelette$
$0{-}omelette \leftarrow rotten, in{-}omelette$
$6{-}omelette \leftarrow fresh, in{-}cup$
$5{-}omelette \leftarrow rotten, in{-}cup$
$\neg wash \leftarrow \text{not } in{-}cup$
$wash \leftarrow in{-}cup$

For the different omelettes we must state that they are mutually inconsistent. We omit the 6 rules necessary for representing this. They are of the form $\neg x{-}omelette \leftarrow y{-}omelette$ with $x \neq y$. We finally represent our desires:

$\neg wash \times wash$
$6{-}omelette \times 5{-}omelette \times 0{-}omelette$

This logic program has the following 6 answer sets

$S_1 = \{6{-}omelette, \neg wash, fresh, in{-}omelette\}$
$S_2 = \{0{-}omelette, \neg wash, rotten, in{-}omelette\}$
$S_3 = \{6{-}omelette, wash, fresh, in{-}cup\}$
$S_4 = \{5{-}omelette, wash, rotten, in{-}cup\}$
$S_5 = \{5{-}omelette, \neg wash, fresh, throw{-}away\}$
$S_6 = \{5{-}omelette, \neg wash, rotten, throw{-}away\}$

Fig. 1 illustrates the preference relationships among answer sets:

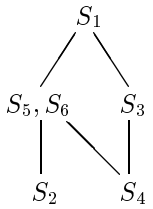

Fig.1: Preferences among answer sets

An optimistic decision maker reasoning from maximally preferred answer sets would choose $in{-}omelette$. A pessimistic decision maker might choose the action whose worst outcome is most tolerable. In the example the answer sets containing $throw{-}away$, that is S_5 and S_6, are preferred to the least preferred answer set containing $in{-}omelette$, S_2, and to the least preferred answer set containing $in{-}cup$, S_4. Thus, a pessimistic decision maker would choose $throw{-}away$. An extremely cautious strategy would prefer a decision C_1 over a decision C_2 if the least preferred answer set(s) containing C_1 are preferred to the most preferred answer set(s) containing C_2. This is a very strong requirement and in the egg example no action is preferred to another one according to this strategy. Finally, we can distinguish a set of state literals Σ and compare answer sets statewise (states are subsets of Σ, the states in the example are $fresh$ and $rotten$). A decision C_1 is preferred over a decision C_2 if for each state $T \subseteq \Sigma$ the least preferred answer set(s) containing $C_1 \cup T$ are preferred to the most preferred answer set(s) containing $C_2 \cup T$.

4 Conclusion

In this paper we introduced a new connective to logic programming. This connective - called ordered disjunction - can be used to represent context dependent preferences in a simple and elegant way. Logic programming with ordered disjunction has interesting applications, in particular in design and configuration, and it can serve as a basis for qualitative decision models. We have constructed a prototype implementation for $LPOD$s based on *Smodels*, an efficient ASP solver developed at Helsinki University of Technology. The prototype implementation is available at http://www.tcs.hut.fi/Software/smodels/priority.

References

[BBL02] G. Brewka, S. Benferhat, and D. Le Berre. Qualitative Choice Logic. In *Proc. Principles of Knowledge Representation and Reasoning, KR-02*. Morgan Kaufmann, 2002. available as IfI-Report under www.informatik.uni-leipzig.de/ brewka.

[ELM+98] T. Eiter, N. Leone, C. Mateis, G. Pfeifer, and F. Scarcello. The KR System dlv: Progress Report, Comparisons and Benchmarks. In *Proc. Principles of Knowledge Representation and Reasoning, KR-98*. Morgan Kaufmann, 1998.

[GL91] M. Gelfond and V. Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New Generation Computing*, 9:365–385, 1991.

[NS97] I. Niemelä and P. Simons. Efficient Implementation of the Stable Model and Well-Founded Semantics for Normal Logic Programs. In *Proc. 4th Intl. Conference on Logic Programming and Nonmonotonic Reasoning*. Springer Verlag, 1997.

[Sav54] L. Savage. *The foundations of Statistics*. Dover, New York, 1954.

[SI94] C. Sakama and K. Inoue. An Alternative Approach to the Semantics of Disjunctive Logic Programs and Deductive Databases. *Journal of Automated Reasoning*, 13:145–172, 1994.