Answer Sets and Qualitative Decision Making

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

Abstract. Logic programs under answer set semantics have become popular as a knowledge representation formalism in Artificial Intelligence. In this paper we investigate the possibility of using answer sets for qualitative decision making. Our approach is based on an extension of the formalism, called logic programs with ordered disjunction ($LPOD$s). These programs contain a new connective called ordered disjunction. The new connective allows us to represent alternative, ranked options for problem solutions in the heads of rules: $A \times B$ intuitively means: if possible A, but if A is not possible then at least B. The semantics of logic programs with ordered disjunction is based on a preference relation on answer sets. We show that $LPOD$s can serve as a basis for qualitative decision making.

1. Introduction

Logic programs under answer set semantics [14] have become popular as a knowledge representation formalism in Artificial Intelligence (AI). There are several reasons for this:

1. On one hand logic programs are expressive enough to model many of the typical knowledge representation problems in AI. For instance, the availability of default negation in the body of rules makes it possible to represent defeasible information.
2. On the other hand, the syntax of logic programs is restrictive enough to allow for efficient implementations, and indeed several highly efficient answer set provers have been implemented, for instance the *Smodels* system developed at Helsinki University of Technology [18] or *dlv* [13] developed at Technical University of Vienna.
3. Answer sets (respectively stable models for programs without classical negation) provide an intuitive semantics for logic programs which avoid the pitfalls of procedural systems like Prolog. For instance, loops in programs are handled properly and results do not depend on the order in which rules are written.
4. It turned out that many problems, for instance in planning and configuration, can be elegantly formulated in a way such that models rather than theorems correspond to problem solutions. If we use such representations in logic programming then the answer sets correspond to models and give the solutions we are looking for [17, 23].

In spite of this success, standard logic programs lack a property which is essential for qualitative decision making: there is at least no easy way of expressing user preferences. For this reason we will use an extension of logic programs which allows us to state such preferences.

In a recent paper [7] a propositional logic called Qualitative Choice Logic (QCL) was introduced. The logic contains a new connective $\times$ representing ordered disjunction, a prioritized version of disjunction. Ordinary disjunction allows us to represent alternative options. For instance, if we plan how to spend a free evening we may use

$$cinema \vee pub \vee tv$$

to describe our different options. Ordered disjunction additionally expresses a preference order among the options:

$$cinema \times pub \times tv$$

intuitively says: if possible I would like to go to the $cinema$, if this is not possible then I go to the pub, if this is also not possible I will watch tv.

The semantics of QCL is based on degrees of satisfaction of a formula in a classical model. The degrees, intuitively, measure disappointment and induce a preference relation on models. Consequence is defined in terms of most preferred models. It is argued in that paper that there are numerous useful applications, e.g. in configuration and design.

The extension of logic programs used in this paper adds ideas underlying QCL to logic programming. More precisely, we use logic programs based on rules with ordered disjunction in the heads. Such programs, called $LPOD$s for short, were first investigated in [6]. The semantics of $LPOD$s is based on a suitable generalization of answer sets together with a preference ordering on the answer sets. The basic intuition is as follows: we use the ordered disjunctions in rule heads to define a preference relation on the answer sets of a program. Consider a program containing the rule

$$A \times B \leftarrow C$$

If S_1 is an answer set containing C and A and S_2 is an answer set containing C and B but not A, then - ceteris paribus (other things being equal) - S_1 is preferred over S_2. Of course, we have to give precise meaning to the ceteris paribus phrase. Intuitively, ceteris paribus is to be read as: S_1 and S_2 satisfy the other rules in the program equally well.

This preference structure on answer sets can be used as a basis for qualitative decision making, as we will demonstrate in this paper. However, for this purpose several additional notions are necessary: in particular, we need to distinguish between choices which are under the control of the agent and possible states which depend on nature and cannot be influenced by the agent. Moreover, we need to fix a strategy towards risk which is used to pick a decision on the basis of the preferences on answer sets..

We will restrict our discussion in this paper to propositional programs. However, as usual in answer set programming, we admit rule schemata containing variables bearing in mind that these schemata are just convenient representations for the set of their ground instances.

The rest of the paper is organized as follows. In the next section we recall the definition of answer sets for extended logic programs with two kinds of negation and give some motivation which may be helpful for readers unfamiliar with this notion. In the subsequent section we define syntax and semantics of $LPOD$s. For a more detailed discussion and additional examples we refer the reader to [6]. Section 4 discusses how to use $LPOD$s for qualitative decision making. We will use Savage's famous rotten egg example to illustrate our approach. Section 5 discusses hybrid decision making systems which are based on a mixed qualitative and quantitative representation of preferences. Section 6 concludes.

2. Answer sets

In this section we recall the definition of answer sets as introduced by Gelfond and Lifschitz [14]. We consider extended logic programs which have two kinds of negation, classical negation $\neg$ and default negation not . Intuitively, $\mathrm{not}\, a$ is true whenever there is no reason to believe a, whereas $\neg a$ requires a proof of the negated literal.

An *extended logic program* (program, for short) P is a finite collection of rules r of the form

$$c \leftarrow a_1, \ldots, a_n, \mathrm{not}\ b_1, \ldots, \mathrm{not}\ b_m \tag{1}$$

where the a_i, b_j and c are classical literals, i.e., either positive atoms or atoms preceded by the classical negation sign $\neg$. We denote by $head(r)$ the head of rule r. We will call $a_1, \ldots, a_n$ the *prerequisites* of the rule and use $pre(r)$ to denote the set of prerequisites of r.

We say a rule r of the form (1) is *defeated by a literal* ℓ, if $\ell = b_i$ for some $i \in \{1, \ldots, m\}$, and we say it is *defeated by a set of literals* X, if X contains a literal that defeats r. Moreover, a rule r is applicable in X whenever it is not defeated by X and its prerequisites are in X.

An answer set of a program P is a set of literals S satisfying two conditions:

1. if $r \in P$ is applicable in S then r is applied, that is, $head(r) \in S$, and
2. all literals in S have a non-circular derivation using only rules undefeated by S.

We can make this precise as follows:

DEFINITION 1. *Let P be an extended logic program, and let X be a set of literals. The X-reduct of P, denoted P^X, is the collection of rules resulting from P by*

1. *deleting each rule which is defeated by X, and*
2. *deleting all weakly negated literals from the remaining rules.*

This reduction is often called Gelfond-Lifschitz reduction, after its inventors.

DEFINITION 2. *Let R be a collection of rules without weak negation. Then, $Cn(R)$ denotes the smallest set S of literals such that*

1. *S is closed under R, i.e., for any rule $c \leftarrow a_1, \ldots, a_n$ in R, if $a_1, \ldots, a_n \in S$, then $a \in S$; and*
2. *S is logically closed, i.e., either S is consistent or $S = Lits$, the set of all literals.*

DEFINITION 3. *Let R be a collection of rules. Define an operator $\gamma_R(X)$ on the set literals as follows:*

$$\gamma_R(X) = Cn(R^X)$$

Then, a set S of literals is an answer set of R iff $S = \gamma_R(S)$.
The collection of answer sets of R is denoted by $AS(R)$.

Here is an example involving both types of negation. The example describes the strategy of a certain college for awarding scholarships to its students. It is taken from [1]:

$$\begin{array}{lll}
(1)\ eligible & \leftarrow & highGPA \\
(2)\ eligible & \leftarrow & minority, fairGPA \\
(3)\ \neg eligible & \leftarrow & \neg fairGPA(x), \neg highGPA \\
(4)\ interview(x) & \leftarrow & \text{not}\, eligible, \text{not}\, \neg eligible
\end{array}$$

Assume in addition to the rules above the following facts about Anne are given:

$$fairGPA, \neg highGPA$$

We obtain exactly one answer set, namely

$$\{fairGPA, \neg highGPA, interview\}$$

Anne will thus be interviewed before a decision about her eligibility is made. If we use the above rules together with the facts

$$minority, fairGPA$$

then the single answer set contains $eligible$.

3. Logic programs with ordered disjunction

In this section we show how ordered disjunction can be added to logic programs with two kinds of negation (default and strong negation) [14]. The new connective $\times$ representing ordered disjunction is allowed to appear in the head of rules only. A (propositional) $LPOD$ thus consists of rules of the form

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

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 $p \leftarrow body, \text{not } p$ for some p not appearing in the rest of the program. The effect of the rule is that no answer sets containing *body* exist.

As mentioned earlier we want to use the ranking of literals in the head of rules to select some of the answer sets of a program as the preferred ones. But what are the answer sets of a program among which to make this selection?

Our semantics is based on so-called split programs. This notion was first introduced in [20]. Our definition is stronger than the one in [20] since we require exactly one option of each rule to be contained in a split program.

DEFINITION 4. *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*

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

DEFINITION 5. *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.*

Here is a simple example. Let P consist of the rules $A \times B \leftarrow \text{not } C$ and $B \times C \leftarrow \text{not } D$. We obtain 4 split programs

$$\begin{array}{ll} A \leftarrow \text{not } C & A \leftarrow \text{not } C \\ B \leftarrow \text{not } D & C \leftarrow \text{not } D, \text{not } B \\ & \\ B \leftarrow \text{not } C, \text{not } A & B \leftarrow \text{not } C, \text{not } A \\ B \leftarrow \text{not } D & C \leftarrow \text{not } D, \text{not } B \end{array}$$

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

DEFINITION 6. *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.*

We exclude inconsistent answer sets from consideration since they do not represent possible problem solutions. In the example above we obtain 3 answer sets: $\{A, B\}, \{C\}, \{B\}$. Note that one of the answer sets is a proper subset of another answer set. Not all of the answer sets satisfy our most intended options. Clearly, $\{B, A\}$ gives us the best options for both rules, whereas $\{C\}$ gives only the second best option for 2) and $\{B\}$ the second best option for 1). To distinguish between more and less intended answer sets we introduce the degree of satisfaction of a rule in an answer set:

DEFINITION 7. *Let S be an answer set of an $LPOD$ P. The satisfaction degree of the rule*

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

in S, denoted $deg^S(r)$ is defined as follows

$$\begin{array}{ll} deg^S(r) = 1 & \textit{if } r \textit{ is not applicable in } S, \\ deg^S(r) = j & \textit{where } j = min\{r \mid C_r \in S\} \textit{ otherwise}. \end{array}$$

It can be shown [6] that an answer set of P satisfies all rules of P to some degree. Degrees can be viewed as penalities: the higher the degree the less satisfactory the answer set. Since inapplicable rules do not lead to any kind of dissatisfaction they get the best possible degree 1.

We use the degrees of satisfaction of a rule to define a preference relation on answer sets. Each rule ranks all answer sets according to its satisfaction degree. From the rankings of the different rules we have to generate a global ranking based on all rules of the program. There are different ways of doing this. For instance, we can simply add up the satisfaction degrees of all rules and prefer those answer sets where the total sum is minimal. Although this may be reasonable in certain applications, this approach makes quite strong assumptions about the commensurability of choices in different rule heads. In [7] a lexicographic ordering of models based on the number of premises satisfied to a particular degree was proposed. This lexicographic ordering has a somewhat syntactic flavour. Therefore, we will use here a 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. For a discussion of alternative preference relations see [9].

DEFINITION 8. *For a set of literals S, let $S^i(P) = \{r \in P \mid deg^S(r) = i\}$. Let S_1 and S_2 be answer sets of P. S_1 is preferred to S_2 ($S_1 > S_2$) iff*

1. *there is an i such that $S_2^i(P) \subset S_1^i(P)$, and*
2. *for all $j < i$, $S_1^j(P) = S_2^j(P)$.*

DEFINITION 9. *A set of literals S is a preferred answer set of an $LPOD$ P iff S is an answer set of P and there is no answer set S' of P such that $S' > S$.*

Consider again the program with rules 1) $A \times B \leftarrow \text{not}\, C$ and 2) $B \times C \leftarrow \text{not}\, D$. As discussed before, we obtain the 3 answer sets: $S_1 = \{A, B\}$, $S_2 = \{C\}$ and $S_3 = \{B\}$. S_1 satisfies both rules with degree 1, $\{C\}$ satisfies 1) to degree 1 but 2) to degree 2. $\{B\}$ satisfies 1) to degree 2 and 2) to degree 1. The single preferred answer set is thus S_1, as intended.

It turns out that $LPOD$s have interesting applications, for instance in configuration and design. $LPOD$s allow us - like normal logic programs - to express incomplete and defeasible knowledge through the use of default negation. In addition, they provide means to represent preferences among intended properties of problem solutions. Moreover, these preferences may depend on the current context. An implementation of $LPOD$s on top of a standard answer set prover for non-disjunctive programs is described in [9].

4. Decision Making using LPODs

In decision making settings it is not sufficient to consider only the most preferred answer sets since this amounts to an extremely optimistic view about how the world will behave (this view is sometimes called wishful thinking). As is well-known in decision theory, for realistic models of decision making it is necessary to clearly distinguish what is under the control of the agent (and thus may constitute the agent's decision) from what is not. We will do this by distinguishing a subset of the literals in a program as decision literals.

In this section we describe a general 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. Let us describe the necessary steps more precisely:

1. Among the literals in the logical language distinguish a set of decision literals C. C is the set of literals the agent can decide upon. It's the agent's decision which makes them true. A decision is a consistent subset of C.
2. Represent the different alternative decisions which can be made by the agent. Standard answer set programming techniques can be used here. Note that certain options may lead to additional choices that need to be made.
3. Represent the different alternative states of the world. Again standard answer set programming techniques apply.
4. Represent relationships between and consequences of different alternatives.

5. Represent desired properties. This is where ordered disjunction comes into play. Of course, desires may be context-dependent.
6. Use the preference relation on answer sets derived form the satisfaction degrees of rules to induce a preference relation on possible decisions. Of course, there are different ways to do this corresponding to different attitudes of the agent towards risk.
7. Pick one of the most preferred decisions.

Let us describe these ideas more formally.

DEFINITION 10. *A decision scenario is a quadruple* $D = (C, S, P, Strat)$ *consisting of*

1. *a set of literals* C *representing possible choices of the agent,*
2. *a set of literals* S *representing possible states of the world,*
3. *an* $LPOD$ P *such that each answer set* A *of* P *has a non-empty intersection with* S *and* C.
4. *a decision strategy* $Strat$*, that is, a function which takes as input a partial order on answer sets and produces a partial order on* C.

DEFINITION 11. *Let* $D = (C, S, P, Strat)$ *be a decision scenario.* $c \in C$ *is an acceptable choice in* D *iff* c *is maximal wrt the partial order* $Strat(AS_P)$ *where* AS_P *is the partial order on the answer sets of* P.

We will use Savage's famous rotten egg example [21] 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

1. add it to the omelette which means the whole omelette may be wasted,
2. throw it away, which means one egg may be wasted, or
3. 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 the example the choices correspond to the actions and the states to the possible states of the egg, that is $C = \{in{-}omelette, in{-}cup, throw{-}away\}$, and $S = \{fresh, rotten\}$. Here are the rules which generate the possible decisions and states of the world:

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

For our example 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:

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

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:

$$
\begin{aligned}
&\neg wash \times wash\\
&6{-}omelette \times 5{-}omelette \times 0{-}omelette
\end{aligned}
$$

This logic program has the following 6 answer sets

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

The preference relation among answer sets is as follows: S_1 is the single maximally preferred answer set. S_5 and S_6 are preferred to S_2 and S_4 but incomparable to S_3. S_3 is preferred to S_4 but incomparable to S_5, S_6 and S_2. S_2 and S_4 are incomparable. Fig. 1 illustrates these relationships:

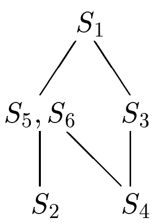

Fig.1: Preferences among answer sets

We next define possible strategies. An optimistic agent might reason from maximally preferred answer sets, that is she might use the strategy defined as

> $c_1 >_o c_2$ *iff the most preferred answer set(s) containing* c_1 *are better than the most preferred answer set(s) containing* c_2.

In the example this would yield $in{-}omelette$ as the single acceptable choice. Obviously, in the example this extremely optimistic attitude towards decision making amounts to simply assuming the egg will be fresh.

A pessimistic decision maker might choose the action whose worst outcome is most tolerable, that is

> $c_1 >_p c_2$ *iff the least preferred answer set(s) containing* c_1 *are better than the least preferred answer set(s) containing* c_2.

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 could be defined as follows:

> $c_1 >_c c_2$ *iff the least preferred answer set(s) containing* c_1 *are better than 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. All possible actions are acceptable.

Finally, we can compare answer sets statewise:

> $c_1 >_{sc} c_2$ *iff for each state* $s \in S$ *the least preferred answer set(s) containing* c_1 *and* s *are better than the most preferred answer set(s) containing* c_2 *and* s.

Also this strategy does not favour any of the choices in the egg example.

In many situations agents consider some of the possible outcomes of a certain choice as completely inacceptable. Assume a set of literals Cat is given characterizing these "catastrophic" outcomes. We can pick any of the strategies $<_k, k \in \{o, p, c, sc\}$ as defined above and add catastrophy avoidance as follows:

> $c_1 >_{k,ca} c_2$ *iff* $c_1 >_k c_2$ *or there is an answer set* S *such that* $c_2 \in S$ *and* $S \cap Cat \neq \emptyset$*, and for each answer set* S', $c_1 \in S'$ *implies* $S' \cap Cat = \emptyset$.

If we consider $0{-}omelette$ as catastropic in the egg example and add catastrophy avoidance to $>_{sc}$ we obtain $throw{-}away$ and $in{-}cup$ as acceptable choices.

Intuitively, S_2 in our example seems far less desirable than S_4 and both S_5 and S_6 less desirable than S_3. This is not reflected in our preference relation on answer sets. To express this it is necessary to represent preferences between sets of literals rather than single literals.

Within our framework this can be done by introducing new atoms representing conjunctions of literals. However, it would probably be more elegant to apply orderd disjunction directly to sets of literals (read as the conjunction of these literals). Extending $LPOD$s in such a way is straightforward.

Another useful extension of $LPOD$s are rankings between the different criteria (that is, rules with ordered disjunction in the heads) themselves. This can be done by splitting a program P into levels $P_1, \ldots, P_n$, where P_i is more important than P_j iff $i < j$. The intuitive idea is to maximize the degree of satisfaction of more important criteria first, and to use less important criteria only to discriminate between those answer sets satisfying the important ones equally well. Rather than presenting the technical details, we want to illustrate this idea using our example. Assume the rule

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

is considered more important than

$$\neg wash \times wash.$$

This leads to a situation where S_3 is preferred over S_5 and S_6. In general, introducing preferences between criteria makes more answer sets comparable which in turn may reduce the number of acceptable choices.

5. Towards hybrid decision making

Classical decision making assumes a complete specification of numerical utility and probability functions. Given these functions maximum expected utility is used to select among the possible actions. In many cases these functions are difficult to obtain, and users are unwilling to come up with quantified specifications of their preferences. This is the main motivation behind qualitative decision making where qualititative preference statements are used instead of numerical functions. On the other hand, there is a price to pay: as illustrated in our treatment of the rotten egg example, a purely qualitative approach is often not fine grained enough to sufficiently discriminate between the available choices.

It is, therefore, a natural idea to explore the middle ground between classical, quantitative approaches on one hand and qualitative approaches on the other. Our long term goal is a hybrid system that is able to handle quantitative information if it is available together with qualitative preference information. The idea is to start with the available mixed information and to compute the set of acceptable choices. If the user is happy with the set, we are done. If she wants further discrimination among this set the system should be able to generate questions about user preferences *which are relevant for discrimination in the current case*. The system would thus be hybrid in a double sense: (a) because it uses qualitative and quantitative information, and (b) because it is a decision making and at the same time a preference elicitation system.

As a first step towards such a system consider an extension of our framework with numerical penalties. We can use integers for this and write, say:

$$\neg wash{-}cup \times wash{-}cup\ (1)$$
$$6{-}omelette \times 5{-}omelette\ (5) \times 0{-}omelette\ (50)$$

to express that having to wash the cup gets penalty 1, getting 5 eggs only penalty 5 and getting nothing to eat penalty 50. The overall penalty for an answer set S is obtained by adding up the penalties for all rules, where the penalty of $c_1 \times c_2(n_2) \times \ldots \times c_k(n_k) \leftarrow$ *body* is 0 if *body* is not satisfied in S or $c_1 \in S$, n_j otherwise, where j is the smallest integer such that $c_j \in S$. The preference relation among answer sets is obtained through their overall penalty. In the example we would obtain the following overall penalties:

$$\begin{array}{lll} S_1 : 0 & S_3 : 1 & S_5 : 5 \\ S_6 : 5 & S_4 : 6 & S_2 : 50 \end{array}$$

Choices could then be ordered on the basis of the average penalties of answer sets they contain. This strategy would thus choose $in{-}cup$ as the single acceptable action.

Every approach to qualitative decision making has to combine preferences among outcomes of choices with a treatment of uncertainty. In our approach the preferences are described through ordered disjunction. But what about the uncertainty? Different possible states of the world are represented as different answer sets. As usual in nonmonotonic reasoning states of the world which are unnormal in some respect are totally disregarded (this is what John McCarthy called jumping to conclusions). All states which have to be taken into account are considered plausible. Further distinctions between the generated answer sets are not possible. For instance, it is not possible to express, say, that $fresh$ is more probable than $rotten$ in the omelette example. If, however the possibility of $rotten$ is negligeable and $fresh$ is true by default we can make sure that only answer sets containing $fresh$ are generated by using adequate rules.

Our general qualitative attitude towards uncertainty can thus be described as: states are either negligeable or plausible; in the latter case no distinction between the degree of plausibility of the states is made.

If additional information about the relative plausibility of the states is available we can take this into account by considering a *weighted* average of penalties, where the weight of an answer set corresponds to the plausibility of the state it contains.

A further elaboration of hybrid decision making is a topic of further study.

6. Conclusion

In this paper we investigated the applicability of logic programs under answer set semantics to qualitative decision making. In particular, we studied programs containing a new connective, ordered disjunction, which can be used to represent context dependent preferences in a simple and elegant way.

There are numerous papers introducing preferences to logic programming. For an overview of some of these approaches see the discussion in [8] or the more recent [22]. Only few of these proposals allow for context dependent preferences. Such preferences are discussed for instance in [5, 8]. The representation of the preferences in these papers is based on the introduction of names for rules, the explicit representation of the preference relation among rules in the logical language, and a sophisticated reformulation of the central semantic notion (answer set, extension, etc.) with a highly self-referential flavour. Alternative approaches [10, 15] are based on compilation techniques and make heavy use of meta-predicates in the logical language. Nothing like this is necessary in our approach. All we have to do is use the degree of satisfaction of a rule to define a preference relation on answer sets directly.

For an overview of recent work in qualitative decision theory see [11]. Poole [19] aims at a combination of logic and decision theory. His approach incorporates quantitative utilities whereas our preferences are qualitative. Interestingly, Poole uses a logic *without* disjunction whereas we *enhance* disjunction.

In [4] CP-networks, a graphical representation, somewhat reminiscent of Bayes nets, for conditional preferences among feature values under the *ceteris paribus* principle is proposed, together with corresponding algorithms. In the boolean case where features have values true or false the graphs correspond to sets of rules of the form

$$a_1, \ldots, a_n : c \succ \bar{c}$$

where c and a_i are literals and $\bar{c}$ is the complement of c. Such a rule has an obvious translation to the $LPOD$ rule

$$c \times \bar{c} \leftarrow a_1, \ldots, a_n$$

which shows that $LPOD$s are syntactically more general: they allow heads of a more general form and default negation in the bodies. However, the semantics of rules in CP-nets is different from the semantics of the translated $LPOD$ rules. Consider the example (from [4]):

$$a \succ \neg a;\ a : b \succ \neg b;\ \neg a : \neg b \succ b$$

In the CP approach $\{a, \neg b\}$ is preferred over $\{\neg a, \neg b\}$. Both sets are answer sets of the translation to $LPOD$s, but none is preferred over the other. The meaning of the rules is slightly different: $a \succ \neg a$ means: world w_1 is better than w_2 if both agree on all atoms except a, and w_1 makes a true, w_2 makes a false. In the $LPOD$ approach $a \times \neg a$ is more like a soft constraint expressing: there is reason to prefer a over $\neg a$. Here an answer set S_1 is preferred over S_2 whenever $a \in S_1$, $\neg a \in S_2$, and the other rules of the program are satisfied at least as well in S_1 as in S_2.

Several models of qualitative decision making based on possibility theory are described in [12, 2]. They are based on certainty and desirability rankings. Some of them make rather strong commensurability assumptions with respect to these rankings. In a series of papers [16, 24], originally motivated by [3], the authors propose viewing conditional desires as constraints on utility functions. Intuitively, $D(a|b)$ stands for: the b-worlds with highest utility satisfy a. Our interpretation of ranked options is very different. Rather than being based on decision theory our approach can be viewed as giving a particular interpretation to the ceteris paribus principle.

Our discussion of hybrid decision making in this paper was quite preliminary. A further elaboration of such systems is a topic of future work. We also plan to investigate generalizations of $LPOD$s. Ordered disjunction requires the literals in the head of a rule to be totally ordered. In many situations total preference information is not available, or different alternatives may be equally preferred. Using ordered disjunction one is forced to introduce arbitrary preferences in such a situation. We will therefore investigate rules whose heads are partially rather than linearly ordered.

In any case, we hope to have convinced the reader that exploring the range between purely qualitative and purely quantitative approaches is promising, and that answer sets of prioritized variants of logic programs may have an interesting role to play in this area.

Acknowledgements

Thanks to S. Benferhat, R. Booth, T. Janhunen, D. Makinson, I. Niemelä, T. Syrjänen and L. van der Torre for helpful comments.

References

1. Baral, C. and M. Gelfond: 1994, 'Logic Programming and Knowledge Representation'. *Journal of Logic Programming* **19/20**, 73–148.
2. Benferhat, S., D. Dubois, H. Fargier, and H. Prade: 2000, 'Decision, Nonmonotonic Reasoning and Possibilistic Logic'. In: J. Minker (ed.): *Logic-Based Artificial Intelligence*. Kluwer Academic Publishers.
3. Boutilier, C.: 1994, 'Towards a Logic for Qualitative Decision Theory'. In: *Proc. Principles of Knowledge Representation and Reasoning, KR-94*. pp. 75–86.
4. Boutilier, C., R. Brafman, H. Hoos, and D. Poole: 1999, 'Reasoning With Conditional Ceteris Paribus Preference Statements'. In: *Proc. Uncertainty in Artificial Intelligence, UAI-99*.
5. Brewka, G.: 1996, 'Well-Founded Semantics for Extended Logic Programs with Dynamic Preferences'. *Journal of Artificial Intelligence Research* **4**, 19–36.
6. Brewka, G.: 2002, 'Logic Programming with Ordered Disjunction'. In: *Proc. AAAI-02*.

7. Brewka, G., S. Benferhat, and D. Le Berre: 2002a, 'Qualitative Choice Logic'. In: *Proc. Principles of Knowledge Representation and Reasoning, KR-02*. available as IfI-Report under www.informatik.uni-leipzig.de/ brewka.
8. Brewka, G. and T. Eiter: 1999, 'Preferred Answer Sets for Extended Logic Programs'. *Artificial Intelligence* **109**, 297–356.
9. Brewka, G., I. Niemelä, and T. Syrjänen: 2002b, 'Implementing Ordered Disjunction Using Answer Set Solvers for Normal Programs'. In: *Proc. 8th European Conference on Logics in Artificial Intelligence (JELIA 2002)*.
10. Delgrande, J., T. Schaub, and H. Tompits: 2000, 'Logic Programs with Compiled Preferences'. In: *Proc. European Conference on Artificial Intelligence, ECAI-00*. pp. 464–468.
11. Doyle, J. and R. Thomason: 1999, 'Background to Qualitative Decision Theory'. *AI Magazine* **20(2)**, 55–68.
12. Dubois, D., D. L. Berre, H. Prade, and R. Sabbadin: 1999, 'Using Possibilistic Logic for Modeling Qualitative Decision: ATMS-based Algorithms'. *Fundamenta Informaticae* **34**, 1–30.
13. Eiter, T., N. Leone, C. Mateis, G. Pfeifer, and F. Scarcello: 1998, 'The KR System dlv: Progress Report, Comparisons and Benchmarks'. In: *Proc. Principles of Knowledge Representation and Reasoning, KR-98*.
14. Gelfond, M. and V. Lifschitz: 1991, 'Classical Negation in Logic Programs and Disjunctive Databases'. *New Generation Computing* **9**, 365–385.
15. Grosof, B.: 1999, 'DIPLOMAT: Compiling Prioritized Rules into Ordinary Logic Programs for E-Commerce Applications (Intelligent Systems Demonstration Abstract)'. In: *Proc. AAAI-99*.
16. Lang, J.: 1996, 'Conditional Desires and Utilities - An Alternative Logical Approach to Qualitative Decision Theory'. In: *Proc. 12th European Conference on Artificial Intelligence, ECAI-96*. pp. 318–322.
17. Lifschitz, V.: 2002, 'Answer Set Programming and Plan Generation'. *Artificial Intelligence Journal* **138(1-2)**, 39–54.
18. Niemelä, I. and P. Simons: 1997, '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*.
19. Poole, D.: 1997, 'The Independent Choice Logic for Modelling Multiple Agents under Uncertainty'. *Artificial Intelligence Journal* **94(1-2)**, 7–56.
20. Sakama, C. and K. Inoue: 1994, 'An Alternative Approach to the Semantics of Disjunctive Logic Programs and Deductive Databases'. *Journal of Automated Reasoning* **13**, 145–172.
21. Savage, L.: 1954, *The foundations of Statistics*. Dover, New York.
22. Schaub, T. and K. Wang: 2001, 'A Comparative Study of Logic Programs With Preference'. In: *Proc. Intl. Joint Conference on Artificial Intelligence, IJCAI-01*.
23. Soininen, T.: 2000, 'An Approach to Knowledge Representation and Reasoning for Product Configuration Tasks'. Ph.D. thesis, Helsinki University of Technology, Finland.
24. van der Torre, L. and E. Weydert: 2001, 'Parameters for utilitarian desires in a qualitative decision theory'. *Applied Intelligence* **14**, 285–301.