Dynamic Decision-Making in Logic Programming and Game Theory

Marina De Vos and Dirk Vermeir

[1] Dept of Computer Science
University of Bath
mdv@cs.bath.ac.uk
[2] Dept. of Computer Science
Vrije Universiteit Brussel, VUB
dvermeir@vub.ac.be

Abstract. We present a framework for decision making with circumstance-dependent preferences and decisions. This formalism, called Ordered Choice Logic Programming, allows decisions that comprise multiple alternatives, which become available only when a choice between them is forced. The skeptical semantics is based on answer sets for which we provide a fixpoint characterization and a bottom-up algorithm. OCLPs can be used to represent and extend game theory concepts. We demonstrate that OCLPs allow an elegant translation of finite extensive games with perfect information such that the c-answer sets correspond to the Nash equilibria of the game. These equilibria are not player-deterministic, in the sense that a single player, given the other players' actions, could rationally leave an equilibrium state by changing her action profile. Therefor cautious Nash equilibria are introduced as the answer sets of the transformed game.

1 Introduction

Preferences or order among defaults and alternatives for a decision play an important role in knowledge representation and non-monotonic reasoning. In case of conflict, humans tend to prefer a default or alternative that corresponds to more reliable, more complete, more preferred or more specifi c information. In recent years, several proposals for the explicit representation of preference in logic programming formalisms have been put forward: [LV90,AAP+98] are just two examples.

Working with preferences/order has applications in various domains, e.g. law, object orientation, model based diagnosis or confi guration tasks. In this paper we present a formalism that enables reasoning about decisions involving multiple alternatives that are dependent on the situation. The dynamics of our formalism is demonstrated by the following example.

Example 1. This year, the choice for a holiday destination has been reduced to a city trip to London or a fortnight stay in either Spain or Cuba. A weekend London is rather short and Cuba is expensive. With a larger budget however, we could have both a holiday in Cuba and a trip to London. Given these considerations, there are two possible outcomes: we have a small budget and we should opt for Spain, or with a larger budget, we can combine Cuba and London.

In both situations we have that Cuba, Spain and London are alternatives for the choice of a travel destination since to have no summer vacation is not an option. In the first outcome, we simply take the best possible alternative: Spain. Otherwise, we have good reason to take more than one alternative. So we take both Cuba and London.

To allow this kind of reasoning we need two mechanisms: one to set the conditional decisions and one to allow circumstance-dependent preferences for the possible alternatives of the decisions. As argued in [DVV99], choice logic programs are an intuitive tool for representing decision-problems, as the semantics ensures that exactly one alternative is chosen when the condition for the decision is met. For the preferences, we use a multi-alternative generalization of the ideas behind ordered logic programming[LV90]. Our formalism, called Ordered Choice Logic Programming (OCLP), combines the best of these formalisms by defining a strict partial order among choice logic programs, called components. Each component inherits the rules from less specific components. The normal semantics is used until a conflict arises; then the more specific alternative is decided upon. We equip our OCLPs with an answer set semantics to obtain the rational solutions for the represented decision-problem. A fixpoint characterization and an efficient algorithm for our semantics will be provided. Furthermore we demonstrate that a logic program or a choice logic program can be elegantly transformed to a negation-free OCLP such that the stable models of the former are obtained as the answer sets of the latter.

Although ordered choice logic programming can add new viewpoints to the above mentioned application domains, we will focus on a novel application in Game Theory. In [DVV00], it was shown that an extensive game with perfect information can be transformed into an OCLP such that the c-answer sets of the latter correspond to the Nash equilibria of the former. Although these equilibria are very useful for predicting the outcome of a game, it is possible that a player, given the other players' actions, still has a rational choice between multiple outcomes. To overcome this, we introduce cautious Nash equilibria as the answer sets of the transformed programs.

2 Choice Logic Programming

A *Choice Logic Program* [DVV99], CLP for short, is a finite set of rules of the form $A \leftarrow B$ where A and B are finite sets of ground atoms. Intuitively, atoms in A are assumed to be mutually exclusive while B is read as a conjunction (note that A may be empty, i.e. constraints are allowed). The set A is called the head of the rule r, denoted H_r, while B is its body, denoted B_r. In examples, we often use "$\oplus$" to denote exclusive or, while "," is used to denote conjunction.

The *Herbrand base* of a CLP P, denoted $\mathcal{B}_P$, is the set of all atoms that appear in P. An *interpretation* is a consistent[1] subset of $\mathcal{B}_P \cup \neg\mathcal{B}_P$. For an interpretation I, we use I^+ to denote its positive part, i.e. $I^+ = I \cap \mathcal{B}_P$. Similarly, we use I^- to denote the negative part of I, i.e. $I^- = \neg(I \cap \neg\mathcal{B}_P)$. An atom a is *true* (resp. *false*) w.r.t. to an interpretation I for a CLP P if $a \in I^+$ (resp. $a \in I^-$). An interpretation is *total* iff $I^+ \cup I^- = \mathcal{B}_P$. The set of all interpretations is denoted $\mathcal{I}_P$. The positive complement of an interpretation I, denoted $\overline{I}$, equals $\mathcal{B}_P \setminus I^+$.

[1] A set A is consistent iff $A \cap \neg A = \emptyset$.

A rule r in a CLP is said to be *applicable* w.r.t. an interpretation I when $B_r \subseteq I$. Since we are modeling choice, we have that r is *applied* when r is applicable and $|H_r \cap I| = 1$[2]. A *model* is defined in the usual way as a total interpretation that makes every applicable rule applied. A model M is said to be *minimal* if there does not exist a model N such that $N^+ \subset M^+$. For choice logic programs, the stable model[3] and the minimal model semantics coincides.

3 Ordered Choice Logic Programming

An ordered choice logic program [DVV00] is a collection of choice logic programs, called components, each representing a portion of information. The relevance or preciseness of each component with respect to the other components is expressed by a strict pointed partial order[4].

Definition 1. *An **Ordered Choice Logic Program**, or OCLP, is a pair $\langle \mathcal{C}, \prec \rangle$ where $\mathcal{C}$ is a finite set of choice logic programs, called **components**, and "$\prec$" is a strict pointed partial order on $\mathcal{C}$.*

For two components $C_1, C_2 \in \mathcal{C}$, $C_1 \prec C_2$ implies that C_2 contains more general information than C_1. Throughout the examples, we will often represent an OCLP P by means of a directed acyclic graph (dag) in which the nodes represent the components and the arcs the $\prec$-relation.

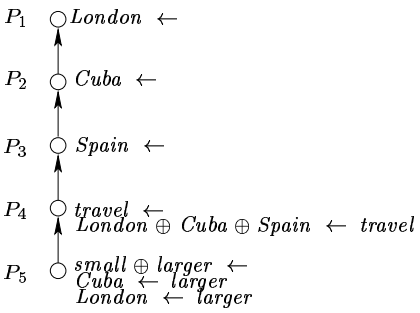

Fig. 1 *The OCLP of Example 2*

Example 2. The decision problem from the introduction (Example 1) can easily be written as the OCLP in Figure 1.

For an OCLP P, we introduce P^* as the CLP that contains all the rules that appear in one of the components of the OCLP. Each rule $r \in P^*$ is assumed to be labeled by the component it was taken from and we use $c(r)$ to retrieve this component.

Having P^*, we can define an interpretation for an OCLP as an interpretation of the underlying P^*, leaving the definitions for an applicable and applied rule unchanged.

Example 3. The sets $I = \{Cuba, small, \neg Spain, \neg travel\}$, $J = \{travel, Cuba, small, \neg London, \neg Spain, \neg larger\}$, $K = \{travel, Spain, small, \neg larger, \neg London, \neg Cuba\}$ and $L = \{travel, larger, London, Spain, Cuba, \neg small\}$ are all interpretations for the OCLP of Example 2. The interpretation I makes the rule $small \oplus larger \leftarrow$ applied while the rule $London \leftarrow$ is applicable but not applied. While J, K and L are total, I is not.

[2] For a a set X, we use $|X|$ do denote its cardinality.

[3] [DVV99] for detailed information.

[4] A relation R on a set A is a strict partial order iff R is anti-reflexive, anti-symmetric and transitive. R is pointed if there is an element $a \in A$ such that aRb for all $b \in A$.

A decision involves a choice between several alternatives, as indicated by so-called *choice rules*, i.e. rules with multiple head atoms. To determine which atoms are alternatives for each other, we also need to take into account the preference order: an atom a is an *alternative* for an atom b in a component C if there is an applicable choice rule present in a component at least as preferred as C, such that a and b appear together in the head.

Definition 2. *Let $P = \langle \mathcal{C}, \prec \rangle$ be an OCLP, I an interpretation and let $C \in \mathcal{C}$. The set of **alternatives in** C for an atom $a \in \mathcal{B}_{P^*}$ **w.r.t.** I, denoted $\Omega^I_C(a)$, is defined as*[5]*:*

$$\Omega^I_C(a) = \{b \mid \exists r \in P^* \cdot c(r) \preccurlyeq C \ \wedge \ B_r \subseteq I \ \wedge \ a, b \in H_r \text{ with } a \neq b\} \ .$$

As long as we do not encounter any conflict, we can adapt the usual ordered logic semantics [LV90] where defeat among rules is used to select a more preferred alternative. But what happens if two alternatives are equally preferred? In this paper, we adopt a cautious approach[6] where a rule r is only defeated if there are more preferred rules suggesting alternatives for each atom in H_r.

Definition 3. *Let I be an interpretation for an OCLP P. A rule $r \in P^*$ is **defeated w.r.t.** I iff $\forall a \in H_r \cdot \exists r' \in P^* \cdot c(r') \prec c(r) \ \wedge \ B_{r'} \subseteq I \ \wedge \ H_{r'} \subseteq \Omega^I_{c(r)}(a)$.*

The reason for requiring that the head of a defeater contains only alternatives makes sure that the defeater is operational in the same context as the defeated rule.

Example 4. Reconsider the interpretations from Example 3. The alternatives for $Cuba$ in P_2 w.r.t. J are $\Omega^J_{P_2}(Cuba) = \{Spain, London\}$. W.r.t. I we obtain $\Omega^I_{P_2}(Cuba) = \emptyset$, since the choice rule in P_3 is not applicable. When we take P_5 instead of P_2, we obtain, w.r.t. J: $\Omega^J_{P_2}(Cuba) = \emptyset$.

The rule $London \leftarrow$ is defeated w.r.t. J by the rule $Cuba \leftarrow$. The rule $London \oplus Cuba \oplus Spain \leftarrow$ is defeated w.r.t. L by the two rules in P_5.

Definition 4. *Let P be an OCLP. A total interpretation I is a **model** iff every rule in P^* is either not applicable, applied or defeated w.r.t. I. A model M is called **minimal** iff M is minimal according to set inclusion.*

Example 5. In Example 3, only K and L are models. Model L is not minimal because of the model $Z = \{travel, larger, Cuba, London, \neg Spain, \neg small\}$. The minimal models K and Z correspond to the intuitive outcomes of the problem.

4 The Answer Set Semantics

4.1 Definition

The simple minimal semantics presented above does not always yield intuitive outcomes, as demonstrated by the program below.

[5] $\preccurlyeq$ is the reflexive closure of $\prec$.

[6] A credulous approach to this program can be found in [DVV00].

Example 6. Consider the program $P = \langle\{c_1, c_2, c_3\}, \prec\rangle$ where $c_1 = \{a \leftarrow\}$, $c_2 = \{b \leftarrow\}$, $c_3 = \{a \oplus b \leftarrow c\}$ and $c_3 \prec c_2 \prec c_1$. The minimal models are $\{a, b\}$, where no choice between a and b is forced, and $\{c, b\}$. The latter is not intuitive due to the gratuitous assumption of c.

We introduce the so-called answer set semantics which, while preserving minimality, prevents unnatural models such as the one from Example 6.

Definition 5. *Let M be a total interpretation for an OCLP P. The **Gelfond-Lifschitz transformation** for P w.r.t. M, denoted P^M, is the choice logic program obtained from P^* by removing all defeated rules. M is called an **answer set** for P iff M is a stable model for P^M.*

4.2 Fixpoint Characterization

Although negation is not explicitly present in an ordered choice logic program, it does appear implicitly. Taking a decision implies that you select one alternative to be true while the others need to be falsifi ed. To group all atoms that may be considered false, we extend the notion of unfounded set for choice logic programs[DVV99] to handle preference. In order to do so, the notion of indefeasible rules is introduced to guarantee that a rule will not be defeated if one extends the current interpretation.

Definition 6. *Let I be an interpretation for an OCLP P. A rule r from P is said to be **indefeasible** w.r.t. I iff r is not defeated w.r.t. any interpretation J such that $I \subseteq J$. A set $X \subseteq \mathcal{B}_{P^*}$ is called an **unfounded set** w.r.t. I iff for each $a \in X$ one of the following conditions is satisfied:*

1. *$\exists r \equiv (a \oplus A \leftarrow B) \in P^* \cdot B \subseteq I \wedge A \cap I \neq \emptyset \wedge r$ is indefeasible w.r.t. I; or*
2. *$\exists \leftarrow B, a \cdot B \subseteq I$; or*
3. *$\forall r \in P^*$ where $a \in H_r$, one of the following conditions holds:*
   (a) $B_r \cap \neg I \neq \emptyset$; or
   (b) $B_r \cap X \neq \emptyset$; or
   (c) r is defeated w.r.t. I; or
   (d) $(H_r \setminus \{a\}) \cap I \neq \emptyset$; or
   (e) $H_r \cap B_r \neq \emptyset$.

*The set of all unfounded sets for P w.r.t. I is denoted $\mathcal{U}_P(I)$. The **greatest unfounded set** for P w.r.t. I, denoted $\mathcal{GUS}_P(I)$, is the union of all unfounded sets for P w.r.t. I. I is said to be **unfounded-free** iff $I \cap \mathcal{GUS}_P(I) = \emptyset$.*

Condition (1) above expresses that the choice is exclusive (r cannot be defeated, so $|H_r \cap I|$ has to be 1 in order for I to be or become a model) and thus alternatives to the actual choice are to be considered false. Condition (2) implies that any atom that would cause a constraint to be violated must be considered false. Condition (3) generalizes the defi nition in [DVV99] where we have added conditions c) and d). The latter is a weaker version of condition (1). In case condition (3) is satisfi ed, we know that there is no reason to consider a true.

It is easy to verify that $\mathcal{GUS}_P(I)$ is itself an unfounded set for P w.r.t. I and that the $\mathcal{GUS}_P$-operator is monotonic.

The greatest unfounded set is a useful tool for the detection of models and answer sets, as demonstrated by the following theorem.

Theorem 1. *Let M be a model for an OCLP P. Then $M^- \in \mathcal{U}_P(M)$. Moreover, M is an answer set iff M is unfounded-free, i.e. $M \cap \mathcal{GUS}_P(M) = \emptyset$, which is itself equivalent to $\mathcal{GUS}_P(M) = M^-$.*

While the $\mathcal{GUS}_P$-operator yields false atoms, the next operator produces atoms that must be true in any model extension of its argument. Combined with $\mathcal{GUS}_P$, we obtain an operator that gives an indication on how to change an interpretation in order to extend it to a model.

Definition 7. *The **immediate consequence operator** $\mathcal{T}_P : 2^{\mathcal{B}_{P^*} \cup \neg\mathcal{B}_{P^*}} \rightarrow 2^{\mathcal{B}_{P^*}}$, where P be an OCLP, is defined by $\mathcal{T}_P(I) = \{a \in \mathcal{B}_{P^*} \mid \exists A \oplus a \leftarrow B \in P^* \cdot A \subseteq \neg I \wedge B \subseteq I \wedge r$ is indefeasible w.r.t. $I\}$.*
The operator $\mathcal{W}_P : \mathcal{I}_P \rightarrow 2^{\mathcal{B}_{P^*} \cup \neg\mathcal{B}_{P^*}}$ is defined by $\mathcal{W}_P(I) = \mathcal{T}_P(I) \cup \neg\mathcal{GUS}_P(I)$.

Theorem 2. *Let P be an OCLP. A total interpretation $M \in \mathcal{I}_P$ is an answer set iff M is a fixpoint of $\mathcal{W}_P$.*

The least fixpoint $\mathcal{W}_P^\omega$ of $\mathcal{W}_P$, if it exists[7], can be regarded as the "kernel" of any answer set (e.g $\mathcal{W}_P^\omega$ is a subset of every answer set). If $\mathcal{W}_P^\omega$ does not not exist, we know that the program does not have an answer set. If $\mathcal{W}_P^\omega$ is total, it must be the unique answer set of P

4.3 Algorithm

For $\mathcal{W}_P$ to be useful in the computation of answer sets, we need a way to compute greatest unfounded set. The best way of dealing with this is providing a fixpoint-operator. This operator, given a program, an interpretation and a set of atoms, should maintain those atoms that can belong to an unfounded set w.r.t. the given interpretation. By repeating this process, starting from the Herbrand base, one automatically obtains the greatest unfounded set. Doing this, we immmediately have a tool to verify unfounded-freeness of an interpretation. The operator taking care of this is called the $\mathcal{R}_{P,I}$-operator. The selection of atoms from the input is identical to verifying whether the input would be an unfounded set, with this difference that instead of having a yes/no answer the operator returns those atoms in the set fulfilling the conditions.

If $\mathcal{W}_P^\omega$ is total, it is the unique answer set. Otherwise, a mechanism is required to proceed from $\mathcal{W}_P^\omega$ toward an answer set. Since $\mathcal{W}_P^\omega \cup \neg\overline{\mathcal{W}_P^\omega}$ cannot be a model, we know that there must exist an applicable rule which is not defeated and not yet applied. In this case we have to choose which of the head elements we will assume to be true; the others will then be assumed false. The combination of such literals is called a choice set. The collection of all these choice sets w.r.t. an interpretation I is denoted $\mathcal{C}_P(I)$. Thus we can go from $\mathcal{W}_P^\omega$ to any answer set by simply adding choice sets until there are no more choice sets available. There is no telling which choice sets should be taken,

[7] The fixpoint may not exists because $\mathcal{W}_P^n(I)$ can become inconsistent, i.e outside of the domain of $\mathcal{W}_P$, for some $n > 0$.

```
Procedure Compute-Answer(I_n:SetOfLiterals);  (* I_n always consistent *)
var X, I'_n, I_{n+1} : SetOfLiterals;
begin
    if C_P(I_n) = ∅  (* no choices available *)
    then if R^ω_{P,I_n ∪ ¬Ī_n}(I_n^+) = ∅ (* and unfounded-free *)
            output "I_n ∪ ¬Ī_n is an answer set of P";
         end-if;
    else for each X ∈ C_P(I_n) do  (* branch over all choice sets *)
                I_{n+1} := I_n ∪ X;  (* Assume the truth of a choice set *)
                repeat  (* add atoms by means of the T_P-operator *)
                    I'_n := I_{n+1};
                    I_{n+1} := T_P(I_n) ∪ I_n;
                until I_{n+1} = I'_n or I_{n+1} ∩ ¬I_{n+1} ≠ ∅;
                if I_{n+1} ∩ ¬I_{n+1} = ∅  (* I_{n+1} is consistent *)
                then Compute-Answer(I_{n+1});
                end-if;
      end-for;
    end-if;
end-procedure;

var I,J : SetOfLiterals;
    G : SetOfAtoms;
begin (*Main *)
    I := ∅;
    repeat  (* Computation of W^ω_P if it exists *)
        J := I;
        G := GUS_P(J);  (* by means of R^ω_{P,J}(B_{P*}) *)
        if G ∩ J ≠ ∅  (* J not unfounded-free *)
        then exit
        end-if;
        I := T_P(J) ∪ ¬G;  (* = W_P(J) *)
    until I = J;
    if I^+ ∪ I^- = B_{P*}
    then output "I is the unique answer set for P";
    else Compute-Answer(I)
    end-if;
end.
```

Fig. 2 *The Algorithm for the computation of answer sets.*

so we need to branch over all of them. To prevent too much wild guessing, we will use a combination of applying choice sets and the immediate consequence operator. Because it is possible that a wrong choice is made, we also need some consistency testing along the way and an unfounded-freeness test of the final interpretation made total by adding the negation of any remaining undecided atoms. An answer set is found if an interpretation survives all the tests. The algorithm is depicted in Fig. 2. For a finite OCLP this program halts in a finite amount of time returning the answer sets.

5 Logic Programming in OCLP

In [DVV99] it was shown that choice logic programs can represent semi-negative logic programs However, the stable models of the logic program did not exactly match with the stable models of the CLP: an extra condition, namely rationality, was required. Generalizing to OCLP, we do obtain a full one-to-one correspondence.

We can take it even a step further to the answer set semantics of general logic programs[8].

Definition 8. *Let P be a general semi-negative logic program. The corresponding OCLP P_L is defined by $\langle\{C, R, N\}, C \prec R \prec N\rangle$ with*

$$\begin{aligned} N &= \{a^\neg \leftarrow \ \mid a \in \mathcal{B}_P\} \ , \\ R &= \{a \leftarrow B, C^\neg \in R \mid r : a \leftarrow B, \neg C \in P\} \cup \\ &\quad \{a^\neg \leftarrow B, C^\neg \in R \mid r : \neg a \leftarrow B, \neg C\} \cup \\ &\quad \{\ \leftarrow B, C^\neg \in R \mid r : \ \leftarrow B, \neg C\} \ , \\ C &= \{a \oplus a^\neg \leftarrow a \mid a \in \mathcal{B}_P\} \ , \end{aligned}$$

where, for $a \in \mathcal{B}_P$, $a^\neg$ is a fresh atom representing $\neg a$.

Intuitively, the choice rules in C force a choice between a and $\neg a$ while the rules in N encode "negation by default" and the rules in R ensure consistency.

Theorem 3. *Let P be a general logic program. Then, $M \subseteq \mathcal{B}_P$ is an answer set of P iff S is an answer set for P_L with*[9]*. $S^+ = M^+ \cup (\mathcal{B}_P \setminus M)^\neg$.*

The proof of this theorem relies on the choice rules to be of the form $a \oplus a^\neg \leftarrow a$. The next example demonstrates that this is indeed essential.

Example 7. Consider the very simple logic program P: $vacation \ \leftarrow \ vacation$. Obviously, we obtain $\emptyset$ as the only answer set of this program. When we apply the transformation of Definition 8, we obtain a OCLP P_L with a single answer set M with $M^+ = \{vacation^\neg\}$. Suppose we would use choice rules with empty body. Then the program P_L would produce two answer sets: M and N with $N^+ = \{vacation\}$.

6 Extensive Games with Perfect Information

An extensive game[OR96] is a detailed description of a sequential structure representing the decision problems encountered by agents (called *players*) in strategic decision making (agents are capable of reasoning about their actions in a rational manner). The agents in the game are perfectly informed of all events that previously occurred. Thus, they can decide upon their action(s) using information about the actions which have already taken place. This is done by means of passing *histories* of previous actions to the deciding agents. *Terminal histories* are obtained when all the agents/players have

[8] Programs that also allow negation-as-failure in the head of their rules [Lif00]

[9] For a set $X \in \mathcal{B}_P$, $X^\neg = \{a^\neg \mid a \in X\}$.

made their decision(s). Players have a preference for certain outcomes over others. Often, preferences are indirectly modeled using the concept of *payoff* where players are assumed to prefer outcomes where they receive a higher payoff.

Summarizing, an extensive game with perfect information is a 4-tuple $\langle N, H, P, (\geq_i)_{i \in N} \rangle$, containing the players N of the game, the histories H, where each history is a sequence of actions, a player function $P : H \rightarrow N$ telling who's turn it is after a certain history and a preference relation $\leq_i$ for each player i over the set of terminal histories. We will use $A(h)$ to denoted the set of actions a player can choose from after an non-terminal history h.

For the examples, we use a more convenient tree representation: each path starting at the root represents a history. The terminal histories are the paths ending in the leafs. The numbers next to the nodes represent the players while the labels on the arcs represent actions. The numbers below the terminal histories are payoffs representing the players' preferences (the first number is the payoff of the first player, the second number is the payoff of the second player).

A *strategy* of a player in an extensive game is a plan that specifies the actions chosen by the player for every history after which it is her turn to move. A *strategy profile* contains a strategy for each player.

The first solution concept for an extensive game with perfect information ignores the sequential structure of the game; it treats the strategies as choices that are made once and for all before the actual game starts. A strategy profile is a *Nash equilibrium* if no player can unilaterally improve upon her choice. Put in another way, given the other players' strategies, the strategy stated for the player is the best she can do.

Example 8. The game in Fig. 3 models an individuals' predicament in the following situation: two ladies have decided that they wanted fruit cake for dessert. There are two possibilities: they either bake a cake or they buy one. At the bakery shop one can choose strawberry and cherry cake. For strawberry cake there is the possibility to have whipped cream on top or not. They agree that the first lady will decide on how to get the cake and, if necessary, whether a topping is wanted or not. The second lady will be picking the type of fruit cake. This game has two Nash equilibria: $\{\{buy, cream\}, \{strawberries\}\}$ and$\{\{buy, no_cream\}, \{cherries\}\}$.

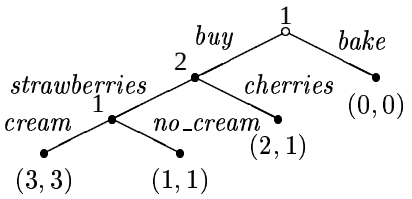

Fig. 3 *The cake game of Example 8*

In [DVV00], it was shown that OCLPs an be used to represent this type of game in order to obtain the equilibria. However, the correspondence relies on a more credulous notion of defeating, which we reproduce below.

Definition 9. *Let I be an interpretation for a OCLP P. A rule $r \in P^*$ is **c-defeated w.r.t.** I iff $\forall a \in H_r \cdot \exists r' \in P^* \cdot c(r) \not< c(r') \wedge r'$ is applied w.r.t. $I \wedge H_{r'} \subseteq \Omega^I_{c(r)}(a)$.*

The difference with ordinary defeat is that we here allow that c-defeaters come from the same or an unrelated component. However in return we demand that this c-defeater is applied instead of just applicable.

The definitions for *c-models* and *c-answer sets* are the same (see Definition 4 and Definition 5) as the definitions for ordinary models and answer sets except that c-defeating is used instead of defeating. The credulous semantics can be used to obtain a game's Nash equilibria.

Definition 10. *Let $\langle N, H, P, (\geq_i)_{i \in N} \rangle$ be a extensive game with perfect information. The corresponding OCLP P_n can be constructed in the following way:*

1. $\mathcal{C} = \{C^t\} \cup \{C_u \mid \exists i \in N, h \in Z \cdot u = U_i(h)\}$;
2. $C^t \prec C_u$ *for all* $C_u \in \mathcal{C}$;
3. $\forall C_u, C_w \in \mathcal{C} \cdot C_u \prec C_w$ *iff* $u > w$;
4. $\forall h \in (H \setminus Z) \cdot (A(h) \leftarrow) \in C^t$;
5. $\forall h = h_1 a h_2 \in Z \cdot a \leftarrow B \in C_u$ *with* $B = \{b \in [h]^{10} \mid h = h_3 b h_4, P(h_3) \neq P(h_1)\}$ *and* $u = U_{P(h_1)}(h)$.

The set of components consists of a component containing all the decisions that need to be considered and a component for each payoff. The order among the components follows the expected payoff (higher payoffs correspond to more specific components) with the decision component at the bottom of the hierarchy (the most specific component). Since Nash equilibria do not take into account the sequential structure of the game, players have to decide upon their strategy before starting the game, leaving them to reason about both past and future. This is reflected in the rules: each rule in a payoff component is made out of a terminal history (path from top to bottom in the tree) where the head represents the action taken when considering the past and future created by the other players according to this history. The component of the rule corresponds with the payoff the deciding player would receive in case the history was carried out.

Theorem 4. *Let $G = \langle N, H, P, (\geq_i)_{i \in N} \rangle$ be a finite extensive game with perfect information and let P_n be its corresponding OCLP. Then, s^* is a Nash equilibrium for G iff s^* is a c-answer set for P_n.*

Example 9. Fig 4 depicts the corresponding OCLP P_n of the game in Example 8. This program has two c-answer sets corresponding to the Nash equilibria of the game.

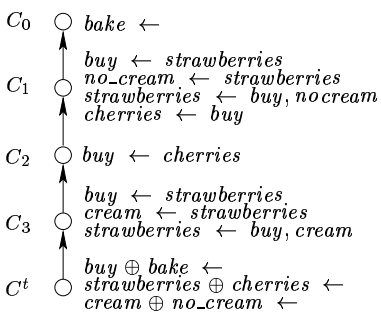

Fig. 4 *P_n of Example 4*

When rational players engage in a game they will always choose strategy profiles that belong to a Nash equilibrium. A deterministic game would be a game for which the outcome was already known from the start. An example of such a game is the Prisoner's Dilemma. Real determinism can only exist when there is a single Nash equilibrium. Demanding that every game should have only one logical outcome would be unrealistic.
However, we feel that having multiple outcomes should be a strategic decision of many

[10] We use $[h]$ to denote the set of actions appearing in a sequence h.

players and not just a single player. In other words, it should not be possible for a single player, given the other players' actions, to deviate rationally from the envisioned outcome. We call this *player-determinism*. The equilibrium $\{\{buy, no\_cream\}, cherries\}$ of Example 8 makes the game not player-deterministic. Given the actions of the first player, e.g. $\{buy, no\_cream\}$, the second player can still rationally decide between $strawberries$ and $cherries$. Both actions would yield her a payoff of 1. So, the first player cannot be certain that she will receive the payoff of Nash equilibrium.

To characterize such situations, we introduce cautious Nash equilibria as the (skeptical) answer sets of the corresponding program.

Definition 11. *Let $G = \langle N, H, P, (\geq_i)_{i \in N} \rangle$ be a finite extensive game with perfect information and let P_n be its corresponding OCLP. A strategy profile s^* is a **cautious Nash equilibrium** iff s^* is an answer set for P_n.*

Example 10. Consider the extensive game from Example 8. This program has one cautious Nash equilibrium: M = $\{\{buy, cream\}, \{strawberries\}\}$.

Theorem 5. *Let $\langle N, H, P, (\geq_i)_{i \in N} \rangle$ be an extensive game with perfect information. Then, every cautious Nash equilibrium for this game is also a Nash equilibrium.*

Cautious Nash equilibria have some interesting properties that distinguish them from normal Nash equilibria. E.g. two cautious Nash equilibria cannot have the same strategies for all but one player unless the same outcome is reached.

7 Relationship with Other Approaches

7.1 OCLP and Game Theory

In the previous section we not only demonstrated that OCLP can be used to retrieve game theoretic notions but that they also allow to extend game theory. Besides defining new equilibria, OCLPs are capable to represent more complex games more easily: variables, depending on the their unification different decisions need to be made and external information influencing the game's outcome are just two examples. Perhaps even more important is the ability for a player to take more than one action as demonstrated by the Travel OCLP of the introduction (Examples 1 and 2). If we just consider the first three components (P_1, P_2 and P_3), we see the representation extensive game with a single player (the person who wants to go on vacation). In this case the equilibrium would be $\{spain\}$ which corresponds to the situation with a smaller budget. With a larger budget, you will be able to afford more than one vacation. In game theory this is simply not possible: every player is forced to take a single action.

7.2 Preferences/Order

Over the years various logic (programming) formalisms have been introduced to deal with the notions of preference, order and updates. Most of these systems can be divided into two groups: the ones that uses the mechanism of preference to filter out

unwanted models and the once that incorporate preference into their model semantics from the start. Examples of the former are: [SI96] with a preference on atoms and [BE99] to obtain the most preferred answer sets. Our OCLPs can found in the latter group. Other examples of such formalisms are: [Bre96] with preferences being part of rules, [BLR98] and [LV90] which use a similar defeating strategy as us for respectively disjunctive logic programs and extended logic programs. The main difference between our systems is the way alternatives are defined. In previous systems alternatives are fixed as an atoms and its (classical) negation. An other example is the dynamic logic programming of [AAP$^+$98]. A stable model of such a dynamic logic program is a stable model of the generalized program obtained by removing the rejected rules. The definition of a rejected rule corresponds to our definition of a defeated rule when a and $\neg a$ are considered alternatives. Since the stable model semantics and the answer set semantics coincide for generalized logic programs. it is not hard to see that Definition 8, with some minor changes, can used to retrieve the stable models of the dynamic logic program. The only thing we need to do is to replace the component R by the P_is of the dynamic logic program $\bigoplus\{P_i : i \in S\}$ and to replace every occurrence of $\neg a$ by $a^{\neg}$.

References

[AAP$^+$98] José Júlio Alferes, Leite J. A., Luís Moniz Pereira, Halina Przymusinska, and Teodor C. Przymusinski. Dynamic logic programming. In Cohn et al. [CSS98], pages 98–111.

[BE99] Gerhard Brewka and Thomas Eiter. Preferred answer sets for extended logic programs. *Artificial Intelligence*, 109(1-2):297–356, April 1999.

[BLR98] Francesco Buccafurri, Nicola Leone, and Pasquale Rullo. Disjunctive ordered logic: Semantics and expressiveness. In Cohn et al. [CSS98], pages 418–431.

[Bre96] Gerhard Brewka. Well-Founded Semantics for Extended Logic Programs with Dynamic Preferences. *Journal of Artificial Intelligence Research*, 4:19–36, 1996.

[CSS98] Anthony G. Cohn, Lenhard K. Schubert, and Stuart C. Shapiro, editors. *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning*, 1998. Morgan Kaufmann.

[DVV99] Marina De Vos and Dirk Vermeir. On the Role of Negation in Choice Logic Programs. In Michael Gelfond, Nicola Leone, and Gerald Pfeifer, editors, *Logic Programming and Non-Monotonic Reasoning Conference (LPNMR'99)*, volume 1730 of *LNAI*, pages 236–246, 1999. Springer Verslag.

[DVV00] Marina De Vos and Dirk Vermeir. A Logic for Modelling Decision Making with Dynamic Preferences. In *Proceedings of the Logic in Artificial Intelligence (Jelia2000) workshop*, number 1999 in *LNAI*, pages 391–406, 2000. Springer Verslag.

[Lif00] Vladimir Lifschitz. Answer set programming and plan generation. *Journal of Artificial Intelligence*, page to appear, 2000.

[LV90] Els Laenens and Dirk Vermeir. A Fixpoint Semantics of Ordered Logic, *Journal of Logic and Computation*, 1(2), pp. 159–185, 1990.

[OR96] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. The MIT Press, Cambridge, Massachusets, London, England, third edition, 1996.

[SI96] Chiaki Sakama and Katsumi Inoue. Representing Priorities in Logic Programs. In Michael Maher, editor, *Proceedings of the 1996 Joint International Conference and Symposium on Logic Programming*, pages 82–96, 1996. MIT Press.