Intention-based Decision Making with Evolution Prospection

Han The Anh and Luís Moniz Pereira

Centro de Inteligência Artificial (CENTRIA)
Departamento de Informtica, Faculdade de Cincias e Tecnologia
Universidade Nova de Lisboa, 2829-516 Caparica, Portugal
h.anh@fct.unl.pt, lmp@di.fct.unl.pt

Abstract. We explore a coherent combination, for decision making, of two Logic Programming based implemented systems, Evolution Prospection and Intention Recognition. The Evolution Prospection system has proven to be a powerful system for decision making, designing and implementing several kinds of preferences and useful environment-triggering constructs. It is here enhanced with an ability to recognize intentions of other agents—an important aspect not explored so far. The usage and usefulness of the combined system is illustrated with several extended examples.

Keywords. Decision Making, Evolution Prospection, Preferences, Intention Recognition, Logic Programming.

1 Introduction

Given the important role that intentions play in the way we make decisions [10,24], one would expect intentions to occupy a substantial place in any theory of action. Surprisingly enough, in what is perhaps the most influential theory of action—rational choice theory—which includes the theory of decision making—explicit reference is made to actions, strategies, information, outcomes and preferences but not to intentions.

This is not to say that no attention has been paid to the relationship between rational choice and intentions. Quite the contrary, a rich philosophical literature has developed on the relation between rationality and intentions (see for example [29]). However, to our knowledge, there has been no real attempt to model and implement the role of intentions in decision making, within a rational choice framework.

In this paper, we set forth a coherent Logic Programming (LP) based system for decision making—which extends the existing work on Evolution Prospection (EP) for decision making [17,18]—but taking into consideration now the intentions of other agents. Obviously, when being immersed in a multi-agent system, knowing the intentions of other agents can benefit the agent in a number of ways. It enable the recognizing agents to predict what other agents will do next or might have done before—thereby, being able to plan in advance and taking the best advantage from the prediction, or acting to take remedial action. In addition, an important role of recognizing intentions is to enable coordination of your own actions and in collaborating with others [11,10]. We have recently studied the role of intention recognition in the evolution of cooperative behavior [14,27], showing that intention recognition strongly promotes the emergence of cooperation in populations of self-regarding individuals [5,4].

The Evolution Prospection system is an implemented LP-based system for decision making [18,22] (described in Section 2). An EP agent can prospectively look ahead a number of steps into the future to choose the best course of evolution that satisfies a goal. This is achieved by designing and implementing several kinds of prior and post preferences and several useful environment-triggering constructs for decision making.

In order to take into account the intentions of other agents in decision making processes, we integrate into EP a previously and separately implemented, but also LP-based, intention recognition system [19,22]. Intention recognition can be defined as the process of inferring the intention or goal of one other agent (called *"individual intention recognition"*) or a group of other agents (called *"collective intention recognition"*) through their observable actions or their actions' observable effects [13,26,1,28]. The intention recognition system performs via Causal Bayesian Networks [15] and plan generation techniques. We will briefly recall the system in Section 3.

2 Evolution Prospection

2.1 Preliminary

The implemented EP system has been proven to be useful for decision making [18]. It has been applied for providing appropriate suggestions for elderly people in Ambient Intelligence domain [2,1]. The advance and easiness of expressing preferences in EP [21,18] enable to closely take into account elders' preferences. The EP system is implemented on top of XSB Prolog [31]. We next describe the constructs of EP, to the extent we use them here. A full account can be found in [18].

Language Let $\mathcal{L}$ be a first order language. A domain literal in $\mathcal{L}$ is a domain atom A or its default negation *not* A. The latter is used to express that the atom is false by default (Closed World Assumption). A domain rule in $\mathcal{L}$ is a rule of the form:

$$A \leftarrow L_1, \ldots, L_t \qquad (t \geq 0)$$

where A is a domain atom and $L_1, \ldots, L_t$ are domain literals. An integrity constraint in $\mathcal{L}$ is a rule with an empty head. A (logic) program P over $\mathcal{L}$ is a set of domain rules and integrity constraints, standing for all their ground instances.

In this paper, we consider solely Normal Logic Programs (NLPs), those whose heads of rules are positive literals, i.e. positive atoms, or empty. We focus furthermore on abductive logic programs, i.e. NLPs allowing for abducibles – user-specified positive literals without rules, whose truth-value is not fixed. Abducibles instances or their default negations may appear in bodies of rules, like any other literal. They stand for hypotheses, each of which may independently be assumed true, in positive literal or default negation form, as the case may be, in order to produce an abductive solution to a query.

Definition 1 (Abductive Solution). *An abductive solution is a consistent collection of abducible instances or their negations that, when replaced by true everywhere in P, affords a model of P that satisfies the query true and the ICs – a so-called abductive model, for the specific semantics being used on P.*

Active Goals In each cycle of its evolution the agent has a set of active goals or desires. We introduce the *on_observe*/1 predicate, which we consider as representing active goals or desires that, once triggered by the observations figuring in its rule bodies, cause the agent to attempt their satisfaction by launching all the queries standing for them, or using preferences to select them. The rule for an active goal AG is of the form:

$$on\_observe(AG) \leftarrow L_1, ..., L_t \ (t \geq 0)$$

where L_1,...,L_t are domain literals. During evolution, an active goal may be triggered by some events, previous commitments or some history-related information. When starting a cycle, the agent collects its active goals by finding all the $on\_observe(AG)$ that hold under the initial theory without performing any abduction, then finds abductive solutions for their conjunction.

Preferring Abducibles An abducible A can be assumed only if it is a considered one, i.e. if it is expected in the given situation, and, moreover, there is no expectation to the contrary

$$consider(A) \leftarrow expect(A),\ not\ expect\_not(A),\ A$$

The rules about expectations are domain-specific knowledge contained in the theory of the program, and effectively constrain the hypotheses available in a situation. To express preference criteria among abducibles, we envisage an extended language $\mathcal{L}^\star$. A preference atom in $\mathcal{L}^\star$ is of the form $a \lhd b$, where a and b are abducibles. It means that if b can be assumed (i.e. considered), then $a \lhd b$ forces a to be assumed too if it can. A preference rule in $\mathcal{L}^\star$ is of the form:

$$a \lhd b \leftarrow L_1, ..., L_t \ (t \geq 0)$$

where $L_1, ..., L_t$ are domain literals over $\mathcal{L}^\star$.

A priori preferences are used to produce the most interesting or relevant conjectures about possible future states. They are taken into account when generating possible scenarios (abductive solutions), which will subsequently be preferred amongst each other a posteriori.

A Posteriori Preferences Having computed possible scenarios, represented by abductive solutions, more favorable scenarios can be preferred a posteriori. Typically, *a posteriori* preferences are performed by evaluating consequences of abducibles in abductive solutions. An *a posteriori* preference has the form:

$$A_i \ll A_j \leftarrow holds\_given(L_i, A_i),\ holds\_given(L_j, A_j)$$

where A_i, A_j are abductive solutions and L_i, L_j are domain literals. This means that A_i is preferred to A_j a posteriori if L_i and L_j are true as the side-effects of abductive solutions A_i and A_j, respectively, without any further abduction when testing for the side-effects. Optionally, in the body of the preference rule there can be any Prolog predicate used to quantitatively compare the consequences of the two abductive solutions.

Evolution Result A Posteriori Preference While looking ahead a number of steps into the future, the agent is confronted with the problem of having several different possible courses of evolution. It needs to be able to prefer amongst them to determine the best courses from its present state (and any state in general). The *a posteriori* preferences are no longer appropriate, since they can be used to evaluate only one-step-far consequences of a commitment. The agent should be able to also declaratively specify preference amongst evolutions through quantitatively or qualitatively evaluating the consequences or side-effects of each evolution choice.

A posteriori preference is generalized to prefer between two evolutions. An *evolution result a posteriori* preference is performed by evaluating consequences of following some evolutions. The agent must use the imagination (look-ahead capability) and present knowledge to evaluate the consequences of evolving according to a particular course of evolution. An *evolution result a posteriori preference* rule has the form:

$$E_i \lll E_j \leftarrow holds\_in\_evol(L_i, E_i), holds\_in\_evol(L_j, E_j)$$

where E_i, E_j are possible evolutions and L_i, L_j are domain literals. This preference implies that E_i is preferred to E_j if L_i and L_j are true as evolution history side-effects when evolving according to E_i or E_j, respectively, without making further abductions when just checking for the side-effects. Optionally, in the body of the preference rule there can be recourse to any Prolog predicate, used to quantitatively compare the consequences of the two evolutions for decision making.

3 Intention Recognition

In [19], a method for individual intention recognition via Causal Bayesian Nets (CBN) and plan generation techniques was presented. The CBN is used to generate conceivable intentions of the intending agent and compute their likelihood conditional on the initially available observations, and so allow to filter out the much less likely ones. The plan generator thus only needs to deal with the remaining more relevant intentions, because they are more probable or credible, rather than all conceivable intentions. In this work we do not need the network causal property; hence, only background of the naive Bayesian Networks is recalled. Note that the first component of the intention recognition system is implemented based on P-log [6,7]—a probabilistic logic framework—implemented on top of XSB Prolog [31]. The second component is also implemented on top of XSB Prolog. This allows us an appropriate and coherent integration of EP and the intention recognition system.

Definition 2. *A Bayes Network is a pair consisting of a directed acyclic graph (DAG) whose nodes represent variables and missing edges encode conditional independencies between the variables, and an associated probability distribution satisfying the Markov assumption of conditional independence, saying that variables are independent of non-descendants given their parents in the graph [16,15].*

Definition 3. *Let G be a DAG that represents causal relations between its nodes. For two nodes A and B of G, if there is an edge from A to B (i.e. A is a direct cause of B), A is called a parent of B, and B is a child of A. The set of parent nodes of a node A is denoted by* parents(A)*. Ancestor nodes of A are parents of A or parents of some ancestor nodes of A. If node A has no parents ($parents(A) = \emptyset$), it is called a* top node*. If A has no child, it is called a* bottom node*. The nodes which are neither top nor bottom are said* intermediate*. If the value of a node is observed, the node is said to be an* evidence node*.*

In a BN, associated with each intermediate node of its DAG is a specification of the distribution of its variable, say A, conditioned on its parents in the graph, i.e. $P(A|parents(A))$ is specified. For a top node, the unconditional distribution of the variable is specified. These distributions are called Conditional Probability Distribution (CPD) of the BN.

Suppose nodes of the DAG form a causally sufficient set, i.e. no common causes of any two nodes are omitted, the joint distribution of all node values of a causally sufficient can be determined as the product of conditional probabilities of the value of each node on its parents $P(X_1, ..., X_N) = \prod_{i=1}^{N} P(X_i|parents(X_i))$, where $V = \{X_i|1 \leq i \leq N\}$ is the set of nodes of the DAG.

Suppose there is a set of evidence nodes in the DAG, say $O = \{O_1, ..., O_m\} \subset V$. We can determine the conditional probability of a variable X given the observed value of evidence nodes by using the conditional probability formula

$$P(X|O) = \frac{P(X, O)}{P(O)} = \frac{P(X, O_1, ..., O_m)}{P(O_1, ..., O_m)} \tag{1}$$

where the numerator and denominator are computed by summing up the joint probabilities over all absent variables with respect to V (see [19] for details).

In short, to define a BN, one needs to specify the structure of the network, its CPD and the prior probability distribution of the top nodes.

Network Structure for Intention Recognition The first phase of the intention recognition system is to find out how likely each conceivable intention is, based on current observations such as observed actions of the intending agent or the effects of its actions had in the environment. A conceivable intention is the one having causal relations to all current observations. It is brought out by using a CBN with nodes standing for binary random variables that represent causes, intentions, actions and effects.

Intentions are represented by those nodes whose ancestor nodes stand for causes that give rise to intentions. Intuitively, we extend Heinze's tri-level model [13,22] with a so-called pre-intentional level that describes the causes of intentions, used to estimate prior probabilities of the intentions. However, if these prior probabilities can be specified without considering the causes, intentions are represented by top nodes (i.e. nodes that have no parents). These reflect the problem context or the intending agent's mental state.

Observed actions are represented as children of the intentions that causally affect them. Observable effects are represented as bottom nodes (having no children). They can be children of observed action nodes, of intention nodes, or of some unobserved actions that might cause the observable effects that are added as children of the intention nodes.

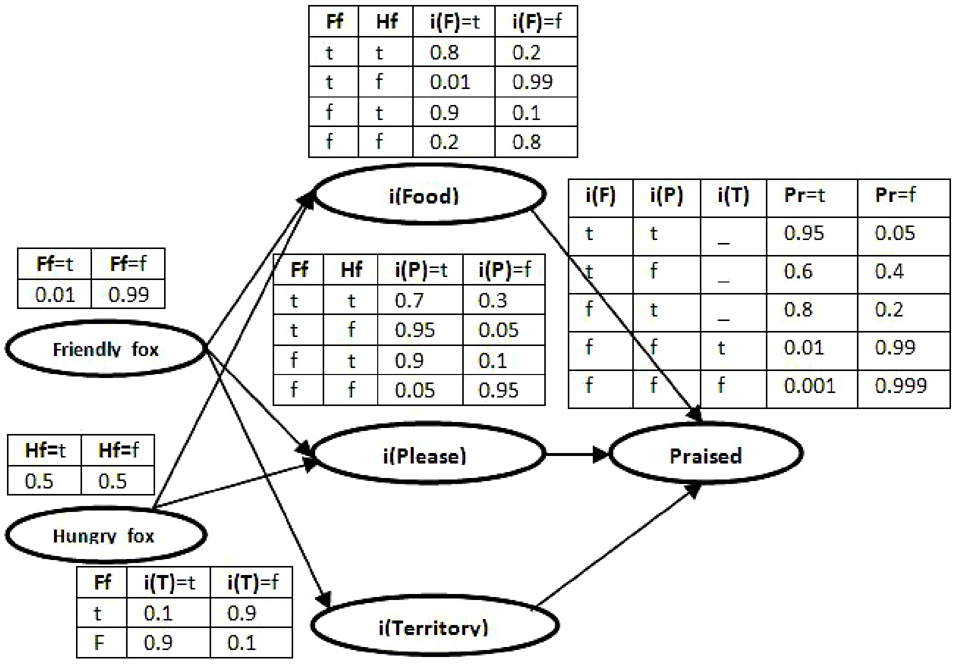
The causal relations among nodes of the CBNs (e.g. which causes give rise to an intention, which intentions trigger an action, which actions have an effect), as well as their Conditional Probability Distribution tables and the distribution of the top nodes, are specified by domain experts. However, they might be learnt mechanically from plan corpora [8,3]. In addition, as it is usually not easy to create the whole BN, we have recently provided a method to incrementally construct it from simple, easily maintained, small fragments of Bayesian Networks [3]. It would enable an easy deployment of the method for real application domains.

Example 1 (Fox-Crow). Consider Fox-Crow story, adapted from Aesop's fable. There is a crow, holding a cheese. A fox, being hungry, approaches the crow and praises her, hoping that the crow will sing and the cheese will fall down near him. Unfortunately for the fox, the crow is very intelligent, having the ability of intention recognition.

The Fox's intentions CBN is depicted in the Figure 1. The initial possible intentions of Fox that Crow comes up with are: Food - $i(F)$, Please - $i(P)$ and Territory - $i(T)$. The facts that might give rise to those intentions are how friendly the Fox is ($Friendly\_fox$) and how hungry he is ($Hungry\_fox$). Currently, there is only one observation which is: Fox praised Crow ($Praised$). Using formula (1) we can compute the probability of each intention conditional on this observation. More details and examples can be found in [22].

4 Evolution Prospection with Intention Recognition

There are several ways an EP agent can benefit from the ability to recognize intentions of other agents, both in friendly and hostile settings. Knowing the in-



Ff	Hf	i(F)=t	i(F)=f
t	t	0.8	0.2
t	f	0.01	0.99
f	t	0.9	0.1
f	f	0.2	0.8

Ff=t	Ff=f
0.01	0.99

Ff	Hf	i(P)=t	i(P)=f
t	t	0.7	0.3
t	f	0.95	0.05
f	t	0.9	0.1
f	f	0.05	0.95

i(F)	i(P)	i(T)	Pr=t	Pr=f
t	t	_	0.95	0.05
t	f	_	0.6	0.4
f	t	_	0.8	0.2
f	f	t	0.01	0.99
f	f	f	0.001	0.999

Hf=t	Hf=f
0.5	0.5

Ff	i(T)=t	i(T)=f
t	0.1	0.9
F	0.9	0.1

Fig. 1: Fox's Intentions CBN

tention of an agent is a means to predict what he will do next or might have done before. The recognizing agent can then plan in advance to take the best advantage of the prediction, or act to take remedial action. Technically, in EP system, this new kind of knowledge may impinge on the body of several EP constructs, such as active goals, expectation and counter-expectation rules, preference rules, context-sensitive integrity constraints, etc., providing a new kind of trigger. In the sequel we draw closer attention to some of those constructs.

4.1 Intention Triggering Active Goals

Recall that an active goal has the form

$$on\_observe(AG) \leftarrow L_1, ..., L_t \ (t \geq 0)$$

where L_1,...,L_t are domain literals. At the beginning of each cycle of evolution, those literals are checked with respect to the current evolving knowledge base and trigger the active goal if they all hold. Now, for intention triggering active goals, the domain literals in the body can be some predicate, either directly or indirectly, affected by intentions of other agents.

It is easily seen that intention triggering active goals are ubiquitous. New goals often appear when we recognize some intentions in others. In a friendly setting, we might want to help others achieve their intention, which is generally represented as follows

```
on_observe(help_achieve_goal(G)) <-
            friend(P), has_intention(P,G).
```

while in a hostile setting, we probably want to prevent the opponents to achieve their goal

```
on_observe(prevent_achieve_goal(G)) <-
            opponent(P), has_intention(P,G).
```

or, perhaps we simply want to plan in advance to take advantage of the hypothetical future obtained when the intending agent employs the plan that achieves his intention

```
on_observe(take_advantage(F)) <-  agent(P),
            has_intention(P,G), future(employ(G),F).
```

Note that the reserved Prolog predicate $has\_intention(P, G)$ holds if person P has an intention or goal G—which is validated by the presented intention recognition system (Section 3). Once a predicate $has\_intention(P, G)$ is called, the (integrated) system triggers its intention recognition component to evaluate if G is the most likely intention of the observed agent (i.e., P). One can also extend to consider N-best intention approach, i.e., assess whether the intention of the agent is amongst the most N likely intentions (see, e.g., our recent work in [3]). Sometimes one needs to be more cautious about the intention of others. Furthermore, it has been shown that by increasing N, the recognition accuracy in significantly improved [9,3]. In general, any intention recognition method can be considered, but to facilitate the integration, LP-based intention recognition method as we adopt here, is most favorable.

Let us look a little closer at each setting, providing some ideas how they can be enacted. When helping someone to achieve an intention, what we need to do is to help him/her with executing a plan achieving that intention successfully, i.e., all the actions involved in that plan can be executed. In contrast, in order to prevent an intention from being achieved, we need to guarantee that all possible plans achieving the intention cannot be executed successfully. For that, at least one action in each plan must be prevented, if the plan is conformant, i.e., a sequence of actions; in case of a conditional plan (see for example [19]), each branch is considered as a conformant plan and must be prevented.

4.2 Intention Triggering Preferences

Having recognized an intention of another agent, the recognizing agent may either favor or disfavor an abducible (*a priori* preferences), an abductive solution (*a posteriori* preferences) or an evolution (*evolution result a posteriori* preferences) with respect to another, respectively, depending on the setting they are in. If they are in a friendly setting, the one which provides more support to achieve the intention is more favored; in contrast, in a hostile setting, the one providing more support is disfavored. The recognizing agent may also favor the one which takes better advantage of the recognized intention.

To illustrate the usage of intention triggering *a priori* preferences, in the sequel we revise the Tea-Coffee example (see [18]).

Example 2 (Tea-Coffee with Intention Recognition). Being thirsty, I consider making tea or coffee. I realize that my roommate, John, also wants to have a drink. To be friendly, I want to take into account his intention when making my choice. This scenario is represented with the following EP program.

```
1. abds([coffee/0, tea/0]).
2. expect(coffee).    expect(tea).
3. on_observed(drink) <- thirsty.
   drink <- tea.      drink <- coffee.
4. expect_not(coffee) <- blood_high_pressure.
5. coffee <| tea <- has_intention(john,coffee).
   tea    <| coffee <- has_intention(john,tea).
```

Fig. 2: Tea-Coffee Considering Intentions

There are two abducibles, *coffee* and *tea*, declared in line 1. Both abducibles are expected (line 2). The only active goal is to *drink*, which is triggered when being thirsty (line 3). The rule in line 4 states that *coffee* is not expected if the blood pressure is high.

In line 5, the first preference says that *tea* is preferable, a priori, to *coffee* if John intends to drink *tea*; and vice versa, if John intends to drink *coffee*, *coffee* is preferable. The recognition of what John intends is done by the intention recognition system described above (Section 3)—which is triggered when a reserved predicate *has_intention*/2 is called.

Next, to illustrate other kind of preferences, consider the following revised extended version of the saving city example, presented in [18].

Example 3 (Saving cities by means of intention recognition). During war time, agent David, a general, needs to decide to save a city from his enemy's attack or leave it to keep the military resource, which might be important for some future purpose. David has recognized that a third party is intending to make an attack to the enemy on the next day. David will have a good chance to defeat the enemy if he has enough military resource to coordinate with the third party. The described scenario is coded with the program in Figure 3.

In the first cycle of evolution, there are two abducibles, *save* and *leave*, declared in line 1, to solve the active goal *choose*—which is triggered when *being attacked* (line 3). Similar to the original version in [18], in the case of being a bad general who just sees the situation at hand, David would choose to *save the city* since

```
1. abds([save/0, leave/0]).
2. expect(save).        expect(leave).
3. on_observe(choose) <- be_attacked.
   choose <- save.                   choose <- leave.
4. save_men(5000) <- save.           save_men(0) <- leave.
   lose_resource  <- save.           save_resource <- leave.
5. Ai << Aj <- holds_given(save_men(Ni), Ai),
            holds_given(save_men(Nj), Aj), Ni > Nj.

6. on_observe(decide)  <-    decide_strategy.
   decide <- stay_still.
   decide <- counter_attack.
7. good_opportunity <-       has_intention(third_party,attack).
   expect(counter_attack) <- good_opportunity, save_resource.
   expect(stay_still).
8. pr(win,0.9)  <- counter_attack.
   pr(win,0.01) <- stay_still.
9. Ei <<< Ej <- holds_in_evol(pr(win,Pi), Ei),
             holds_in_evol(pr(win,Pj), Ej), Pi > Pj.
```

Fig. 3: Saving or Leaving

it would save more people (5000 vs. 0, line 4), i.e. the *a posteriori* preference in line 5 is taken into account immediately, to rule out the case of leaving the city since it would save less people. Then, next day, he would not be able to attack since the military resource is not saved (line 7), and that leads to the outcome with very small probability of winning the whole war (line 8).

But, fortunately, being able to look ahead plus to do intention recognition, David can see that on the next day, if he has enough military resources, he has a good opportunity to make a counter-attack on his enemy (line 7), by coordinating with a third party who exhibits the intention to attack the enemy on that day as well; and a successful counter-attack would lead to a very much higher probability of winning the conflict as a whole (line 8). The *evolution result a posteriori* preference is employed in line 9 to prefer the evolution with higher probability of winning the whole conflict.

In this example we can see, in line 7, how a detected intention of another agent can be used to enhance the decision making process. It is achieved by providing an (indirect) trigger for an abducible expectation which affects the *evolution result a posteriori* preference in line 9.

4.3 Hostile setting

In this hostile setting, having confirmed the intention, and the plans achieving that intention being followed by the intending agent, the recognizing agent must act to prevent those plans from happening, i.e., prevent at least one action of each plan to be successfully executed; and in case of impossibility to do so, act to reduce as much as possible the losses.

Example 4 (Fox-Crow, cont'd). Suppose in Example 1, the final confirmed Fox's intention is that of getting food (to see how it is actually recognized look at ref. [19]). Having recognized Fox's intention, what should Crow do to prevent Fox from achieving it? The following EP program helps Crow with that.

```
1. abds([decline/0, sing/0, hide/2, eat/2, has_food/0, find_new_food/0]).
2. expect(decline). expect(sing). expect(hide(_,_)). expect(eat(_,_)).
3. on_observe(not_losing_cheese) <- has_intention(fox, food).
   not_losing_cheese <- decline.
   not_losing_cheese <- hide(crow,cheese), sing.
   not_losing_cheese <- eat(crow,cheese), sing.
4. expect_not(eat(A,cheese)) <- animal(A), full(A).
   animal(crow).
5. <- decline, sing.    <- hide(crow,cheese), eat(crow,cheese).
6. eat(crow,cheese) <| hide(crow,cheese).
7. no_pleasure <- decline.   has_pleasure <- sing.
8. Ai << Aj <- holds_given(has_pleasure,Ai), holds_given(no_pleasure,Aj).

9. on_observe(feed_children) <- hungry(children).
   feed_children <- has_food.   feed_children <- find_new_food.
   <- has_food, find_new_food.
10.expect(has_food) <- decline, not eat(crow,cheese).
   expect(has_food) <- hide(crow,cheese), not stolen(cheese).
   expect(find_new_food).
11.Ei <<< Ej <- hungry(children), holds_in_evol(had_food,Ei),
                holds_in_evol(find_new_food,Ej).
12.Ei <<< Ej <- holds_in_evol(has_pleasure,Ei),
                holds_in_evol(no_pleasure,Ej).
```

There are two possible ways so as not to lose the Food to Fox, either simply decline to sing (but thereby missing the pleasure of singing) or hide or eat the cheese before singing.

Line 1 is the declaration of program abducibles (the last two abducibles are for the usage in the second phase, starting from line 9). All of them are always expected (line 2). The counter-expectation rule in line 4 states that an animal is not expected to eat if he is full. The integrity constraints in line 5 say that Crow cannot decline to sing and sing, hide and eat the cheese, at the same time. The *a priori* preference in line 6 states that eating the cheese is always preferred to hiding it (since it may be stolen), of course, just in case eating is a possible solution (this is assured in our semantics of *a priori* preferences [18]).

Suppose Crow is not full. Then, the counter expectation in line 4 does not hold. Thus, there are two possible abductive solutions: *[decline]* and *[eat(crow,cheese), sing]* (since the *a priori* preference prevents the choice containing *hiding*).

Next, the *a posteriori* preference in line 8 is taken into account and rules out the abductive solution containing *decline* since it leads to having *no pleasure* which is less preferred to *has pleasure*—the consequence of the second solution that contains *sing* (line 7). In short, the final solution is that Crow eats the cheese then sings, without losing the cheese to Fox and having the pleasure of singing.

Now, let us consider a smarter Crow who is capable of looking further ahead into the future in order to solve longer term goals. Suppose that Crow knows that her children will be hungry later on, in the next stage of evolution (line 9); eating the cheese right now would make her have to find new food for the hungry children. Finding new food may take long, and is always less favourable than having food ready to feed them right away (*evolution result a posteriori* preference in line 11). Crow can see three possible evolutions: $[[decline], [had\_food]]$; $[[hide(crow, cheese), sing], [had\_food]]$ and $[[eat(crow, cheese), sing], [find\_new\_food]]$. Note that in looking ahead at least two steps into the future, local preferences are not taken into account only after all evolution one were applied (full discussion can be found in [18]).

Now the two *evolution result a posterirori* preferences in lines 11-12 are taken into account. The first one rules out the evolution including *finding new food* since it is less preferred than the other two which includes *had food*. The second one rules out the one including *decline*. In short, Crow will hide the food to keep for her hungry children, and still take pleasure from singing.

5 Related Work

Many issues concerning intentions have been widely discussed in the literature of agent research. Some philosophers, e.g., Bratman [10,11] have been concerned with the role that intention plays in directing rational decision making and guiding future actions. Many agent researchers have recognized the importance of intentions in developing useful agent theories, architectures, and languages, such as Rao and Georgeff with their BDI model [23], which has led to the commercialization of several high-level agent languages (e.g., see [12,30]).

However, to the best of our knowledge, there has been no real attempt to model and implement the role of intentions in decision making, within a rational choice framework. Intentions of other relevant agents are always assumed to be given as the input of a decision making process; no system that integrates a real intention recognition system into a decision making system has been implemented so far.

The existent work of Pereira and Han [2,20,22] also attempts to combine the two systems, Evolution Prospection and Intention Recognition, but in a completely different manner. They use an intention recognition system to recognize the goal of the observed agent (e.g., an elder [20]), which the evolution prospection system uses to derive appropriate courses of actions to help achieve. Our approach is more general and genuinely integrated: the intention recognition system is employed also to evaluate other different kinds of information being utilized within an EP program here.

6 Conclusions and Future Work

We have summarized the existent work on Evolution Prospection and Intention Recognition and shown a coherent combination of them for decision making. The Evolution Prospection system has been proven to be a useful one for decision making, and now it has been empowered to take into account intentions of other agents—an important aspect that had not been explored so far. The fact that both systems are LP-based enabled their easy integration. We have described and exemplified several ways in which an Evolution Prospection agent can benefit from having an ability to recognize intentions of other (relevant) agents.

As a future direction, we will apply our combined system to tackle different real application domains, e.g., Ambient Intelligence [2,25] and Elder Care [20], where decision making techniques as well as intention recognition abilities are of increasing importance [26,25,2].

We also plan to provide a formal semantics for our new combined system on top of the one of Evolution Prospection—given in [18]—and the theoretical modelling of intention within a rational choice framework [24].

7 Acknowledgments

We thank the reviewers for useful comments. HTA acknowledges the support from FCT-Portugal, grant SFRH/BD/62373/2009.

References

1. H. T. Anh and L. M. Pereira. Collective intention recognition and elder care. In *AAAI 2010 Fall Symposium on Proactive Assistant Agents (PAA 2010)*. AAAI, 2010. http://www.aaai.org/ocs/index.php/FSS/FSS10/paper/view/2178/2697.
2. H. T. Anh and L. M. Pereira. Proactive intention recognition for home ambient intelligence. In *IE Workshop on AI Techniques for Ambient Intelligence, Ambient Intelligence and Smart Environments*, volume 8, pages 91–100. IOS Press, 2010.
3. H. T. Anh and L. M. Pereira. Context-dependent incremental intention recognition through bayesian network model construction. In A. Nicholson, editor, *Bayesian Modelling Applications Workshop (BMAW-11), Conference on Uncertainty in Artificial Intelligence (UAI-2011)*. CEUR Workshop Proceedings, 2011.
4. H. T. Anh, L. M. Pereira, and F. C. Santos. The role of intention recognition in the evolution of cooperative behavior. In *IJCAI'2011*, 2011.
5. H. T. Anh, L. M. Pereira, and F. C. Santos. Intention recognition promotes the emergence of cooperation. *Adaptive Behavior*, June, 2011.
6. H. T. Anh, C. K. Ramli, and C. V. Damásio. An implementation of extended p-log using xasp. In *Proceedings of International Conference on Logic Programming (ICLP08)*, pages 739–743. Springer LNCS 5366, 2008.

7. C. Baral, M. Gelfond, and N. Rushton. Probabilistic reasoning with answer sets. *Theory and Practice of Logic Programming*, 9(1):57–144, 2009.
8. N. Blaylock and J. Allen. Corpus-based, statistical goal recognition. In *Proceedings of the 18th international joint conference on Artificial intelligence (IJCAI'03)*, pages 1303–1308, 2003.
9. N. Blaylock and J. Allen. Statistical goal parameter recognition. In Shlomo Zilberstein, Jana Koehler, and Sven Koenig, editors, *Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS04)*, pages 297–304. AAAI, 2004.
10. M. E. Bratman. *Intention, Plans, and Practical Reason.* The David Hume Series, CSLI, 1987.
11. M. E. Bratman. *Faces of Intention: Selected Essays on Intention and Agency.* Cambridge University Press, 1999.
12. B. Burmeister, M. Arnold, F. Copaciu, and G. Rimassa. BDI-agents for agile goal-oriented business processes. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: industrial track*, AAMAS '08, pages 37–44, 2008.
13. C. Heinze. *Modeling Intention Recognition for Intelligent Agent Systems.* PhD thesis, The University of Melbourne, Australia, 2003.
14. Martin A. Nowak. Five rules for the evolution of cooperation. *Science*, 314(5805):1560, 2006. DOI: 10.1126/science.1133755.
15. J. Pearl. *Causality: Models, Reasoning, and Inference.* Cambridge U.P, 2000.
16. Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.
17. L. M. Pereira and H. T. Anh. Evolution prospection. In *Proceedings of International Symposium on Intelligent Decision Technologies (KES-IDT'09)*, pages 51–63. Springer Studies in Computational Intelligence 199, 2009.
18. L. M. Pereira and H. T. Anh. Evolution prospection in decision making. *Intelligent Decision Technologies*, 3(3):157–171, 2009.
19. L. M. Pereira and H. T. Anh. Intention recognition via causal bayes networks plus plan generation. In *Progress in Artificial Intelligence, Proceedings of 14th Portuguese International Conference on Artificial Intelligence (EPIA'09)*, pages 138–149. Springer LNAI 5816, October 2009.
20. L. M. Pereira and H. T. Anh. Elder care via intention recognition and evolution prospection. In *select extended papers from the 18th International Conference on Applications of Declarative Programming and Knowledge Management (INAP'09)*, pages 170–187. Springer, LNAI 6547, 2011.
21. L. M. Pereira, P. Dell'Acqua, A. M. Pinto, and G. Lopes. Inspecting and preferring abductive models. In *Handbook on Reasoning-based Intelligent Systems.* World Scientific Publishers, 2011. (forthcoming).
22. L. P. Pereira and H. T. Anh. Intention recognition with evolution prospection and causal bayesian networks. In Ana Madureira, Judite Ferreira, and Zita Vale, editors, *Computational Intelligence for Engineering Systems: Emergent Applications*, volume 46, pages 1–33. Springer, 2011.
23. A. S. Rao and M. P. Georgeff. BDI-agents: from theory to practice. In *Proceeding of First International Conference on Multiagent Systems*, 1995.
24. Olivier Roy. *Thinking before Acting: Intentions, Logic, Rational Choice.* PhD thesis, ILLC Dissertation Series DS-2008-03, Amsterdam., 2009.
25. F. Sadri. Ambient intelligence, a survey. *ACM Computing Surveys*, 2010.
26. F. Sadri. Logic-based approaches to intention recognition. In *Handbook of Research on Ambient Intelligence: Trends and Perspectives.* 2010.

27. Karl Sigmund. *The Calculus of Selfishness*. Princeton U. Press, 2010.
28. Gita Sukthankar and Katia Sycara. Robust and efficient plan recognition for dynamic multi-agent teams. In *Proceedings of International Conference on Autonomous Agents and Multi-Agent Systems*, pages 1383–1388, 2008.
29. M. van Hees and O. Roy. Intentions and plans in decision and game theory. In Bruno Verbeek, editor, *Reasons and intentions*, pages 207–226. Ashgate Publishers, 2008.
30. Michael Wooldridge. Reasoning about rational agents. *The Journal of Artificial Societies and Social Simulation*, 5, 2002.
31. XSB. The XSB system version 3.2 vol. 2: Libraries, interfaces and packages, March 2009.