Defeasible Decision Making in a Multi-robot Environment

Edgardo Ferretti[1], Nicolás Rotstein[2,3], Marcelo Errecalde[1], Alejandro García[2,3], and Guillermo Simari[3]

[1] Laboratorio de Investigación y Desarrollo en Inteligencia Computacional
Universidad Nacional de San Luis, Argentina.
{ferretti,merreca}@unsl.edu.ar
[2] Consejo Nacional de Investigaciones Científicas y Técnicas
[3] Department of Computer Science and Engineering
Universidad Nacional del Sur, Bahía Blanca, Argentina.
{ndr,ajg,grs}@cs.uns.edu.ar

Abstract. In this work, we present a Defeasible Logic Programming approach to decision making in a multi-robot environment. It will be shown how a successful tool for knowledge representation and defeasible reasoning could be applied to the problem of deciding which task should be performed next. Besides, through several examples, we aim to show how flexible is this approach to program the robots' preference policy, considering a simple application domain with real *Khepera* 2 robots.

1 Introduction

Decision making models for autonomous agents have received increased attention, particularly in the field of intelligent robots. The proposed models are often based on formal theories of decision, such as Classical Decision Theory [1], Qualitative Decision Theory [2] and BDI logics [3]. In other cases, models from neuroscience, cognitive psychology and ethology are considered. In these models, the agents' decision making process is an emergent phenomenon of the interaction of elemental behaviors [4]. An established approach to decision making in robotic systems is that of reactive decision systems. In such systems the decision process is usually dedicated to the selection of the action to be executed, based on the current perceptual information with little (if any) pre-processing.

When applicable, the reactive approach has the advantages of simplicity and speed. However, there are domains in which reactive approaches to decision making become exceedingly difficult to apply or may not intuitively describe the behavior desired. In such cases, the agent's decision can be partly determined by immediate perceptual data but may also include a complete history of previous perceptions and decisions. The agent may also need to consider questions such as: *Which is the more appropriate goal to pursue in the current situation?* or *Which one of the alternative plans do I have to select to reach a certain goal?* Further complicating matters are that the information used in the decision process is (in most domains) incomplete and potentially inconsistent.

Advances in Artificial Intelligence and Applications
Research in Computer Science 32, 2007, pp. 150–160

Received 16/06/07
Accepted 31/08/07
Final version 17/09/07

In this paper we will show how a Defeasible Logic Programming approach could be applied in a robotic domain for knowledge representation and reasoning about which task to perform next. At this end, we have selected a simple application domain where real robots perform cleaning tasks. We use the *Khepera* 2 robot [5], a miniature mobile robot ideal for this kind of experimentation.

The experimental environment consists of a square arena of 100 units per side which is conceptually divided into square cells of 10 units per side each. In this environment, more than one robot could be acting at the same time, but there is no communication among them. There is a global camera which provides the necessary information to perform their activities. The *store* is a 30 × 30 units square on the top-right corner and represents the target area where the boxes should be transported. There are boxes of three different sizes (*small*, *medium* and *big*) spread over the environment. At most two boxes can be stacked, but a box cannot be stacked on top of a smaller box. Thus, big boxes are always on the floor. Figure 1 shows four different scenarios of the above-described environment, depicted in a schematic way. Due to space restrictions, in each scenario, only the portion of the arena containing the involved objets is shown.

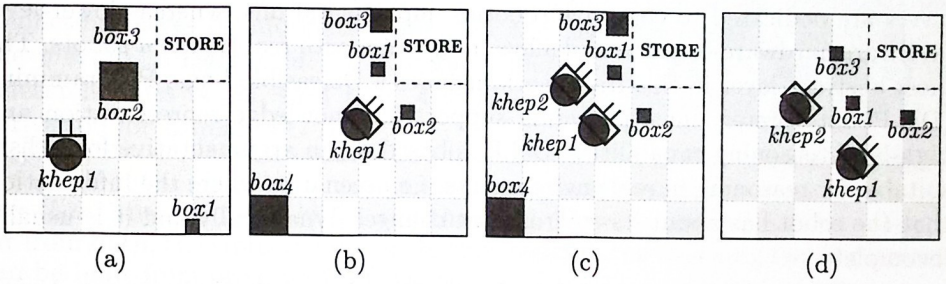

Fig. 1. Four different scenarios

Since in our proposed domain several robots cooperate and there is no communication among them, a robot will not be able to predict exactly the behavior of its partners. Therefore, the robots cannot perform a globally optimized task. In contrast, they will reason about which box is more convenient to select next. To perform the reasoning, a robot will use the perceptual information about the boxes and other robots, and its preferences which will be represented with defeasible rules. For example, a robot could prefer the smallest box, or the nearest one, or the box that is nearest to the store. As we will show below, these preferences will be defeasible and they may change according to the current situation or the presence of other robots. Arguments for and against selecting a box will be considered in order to select the more appropriate one.

A robot capable of solving this kind of problems must at least address the following issues: to perceive the surrounding world, to decide which goal has to be reached, and to have the capabilities for reaching this goal. Several architectures have been proposed in the literature which provide the agents with these skills [6–8]. In this work, we only consider the necessary reasoning processes to make

decisions about which is the most suitable box to be transported by each robot. We are not going to consider problems related to the implementation of low-level actions and sensorial perception. Moreover, some of the aspects related to the sensorial and effectorial support for the Khepera robots have received attention elsewhere [9].

The paper is organized as follows. In Sect. 2, we describe how the robots represent their knowledge and present a succinct overview of the reasoning formalism they use. Then, Sect. 3 shows how the decision process is performed through several examples, in which one or more robots are working together in the environment. Finally, Sect. 4 puts forward the conclusions and related work.

2 Khepera Robots with Argumentative Reasoning

In our approach, knowledge representation and reasoning for decision making will be done using the *Khe*-DeLP framework [9]. *Khe*-DeLP is a layered framework that provides facilities to the Khepera community for programming high-level robots' behavior using an argumentative approach. In *Khe*-DeLP, upper layers are dedicated to cognitive robotics implementations, whereas lower-level layers are hardware-oriented and allow interaction with real *Khepera* robots. The most abstract layer in this framework includes a Defeasible Logic Programming (DeLP) interpreter, thus, providing support for knowledge representation and high-level reasoning capabilities. DeLP is based on an argumentative formalism suitable for reasoning in real environments, *i.e.*, scenarios where the information that the robot has about its environment changes dynamically and it is usually incomplete.

In DeLP (see [10] for a complete description), knowledge is represented with a DeLP-program $\mathcal{P}$ that contains facts, strict rules and defeasible rules. When required, $\mathcal{P}$ is denoted (Π, Δ) distinguishing the subset Π of facts and strict rules, and the subset Δ of defeasible rules. *Facts* are ground literals representing atomic information or the negation of atomic information. In our application examples, robots' perception will be represented with facts, *e.g.*, $nearer\_store(box2, box1)$.

Strict Rules are denoted $L_0 \leftarrow L_1, \ldots, L_n$ and represent firm information, whereas *Defeasible Rules* are denoted $L_0 \prec L_1, \ldots, L_n$ and represent tentative information. In both cases, the *head* L_0 is a literal and the *body* $\{L_i\}_{i>0}$ is a set of literals. As usual in *Logic Programming*, variables are denoted with an initial uppercase letter. Figure 2 shows the DeLP-program that will be used in our application examples. There, for example, the strict rule "$choose(R, B) \leftarrow unique(B)$" represents that "if B is the only box in the environment, then it must be chosen by robot R", and "$\sim choose(R, B) \leftarrow on(Obox, B)$" states that "if B has a box $Obox$ on its top then do not choose B". In the same figure, the defeasible rule "$choose(R, B) \prec better(R, B, Obox)$" states that " if box B is better than box $Obox$, then robot R has a (defeasible) reason for choosing B", whereas "$\sim choose(R, B) \prec better(R, Obox, B)$" states that "if there is other box that is better than B, then there is a reason for not choosing B". Observe that strong

negation ("$\sim$") is allowed in the head of program rules, and hence may be used to represent contradictory knowledge.

$$better(R, B, Obox) \prec nearer\_robot(R, B, Obox) \quad (1)$$
$$better(R, B, Obox) \prec nearest\_robot(R, B) \quad (2)$$
$$better(R, B, Obox) \prec smaller(B, Obox) \quad (3)$$
$$better(R, B, Obox) \prec nearer\_store(B, Obox) \quad (4)$$
$$\sim better(R, B, Obox) \prec nearer\_robot(R, Obox, B) \quad (5)$$
$$\sim better(R, B, Obox) \prec nearest\_robot(R, Obox), diff(B, Obox) \quad (6)$$
$$\sim better(R, B, Obox) \prec nearer\_store(Obox, B) \quad (7)$$
$$\sim better(R, B, Obox) \prec smaller(Obox, B) \quad (8)$$
$$\sim better(R, B, Obox) \leftarrow same\_properties(R, Obox, B) \quad (9)$$
$$\sim better(R, Obox, B) \leftarrow same\_properties(R, Obox, B) \quad (10)$$
$$choose(R, B) \prec better(R, B, Obox) \quad (11)$$
$$\sim choose(R, B) \prec better(R, Obox, B) \quad (12)$$
$$\sim choose(R, B) \prec better(R, B, Obox), better(R, BetterB, B), diff(Obox, BetterB) \quad (13)$$
$$choose(R, B) \prec same\_properties(R, Obox, B) \quad (14)$$
$$\sim choose(R, B) \prec diff\_robot(R, Or), better(Or, B, \_), choose(R, Obox), diff(B, Obox) \quad (15)$$
$$choose(R, B) \leftarrow unique(B) \quad (16)$$
$$\sim choose(R, B) \leftarrow on(Obox, B) \quad (17)$$
$$\sim choose(R, B) \leftarrow carrying(Or, B) \quad (18)$$

Fig. 2. $\mathcal{P}_k = (\Pi_k, \Delta_k)$

In DeLP, to deal with contradictory and dynamic information, *arguments* for conflicting pieces of information are built and then compared to decide which one *prevails*. An *argument* for a literal L, denoted $\langle \mathcal{A}, L \rangle$, is a minimal set of defeasible rules $\mathcal{A} \subseteq \Delta$, such that $\mathcal{A} \cup \Pi$ is non-contradictory and there is a derivation for L from $\mathcal{A} \cup \Pi$.

For instance, let us consider the situation depicted in Fig. 1(a), where there is a single robot ($khep1$), a medium size box ($box3$) near to the store, and a big box ($box2$) near to both $khep1$ and the store. There is also a small box ($box1$) far from both, the store and $khep1$. In this scenario, the following two arguments can be built from program $\mathcal{P}_k$ (Fig. 2):

$$\mathcal{A}_1 = \left\{ \begin{array}{l} choose(khep1, box2) \prec better(khep1, box2, box3) \\ better(khep1, box2, box3) \prec nearer\ robot(khep1, box2, box3) \end{array} \right\}$$

$$\mathcal{A}_2 = \left\{ \begin{array}{l} \sim choose(khep1, box2) \prec better(khep1, box3, box2) \\ better(khep1, box3, box2) \prec smaller(box3, box2) \end{array} \right\}$$

In DeLP, to establish if $\langle \mathcal{A}, L \rangle$ is a non-defeated argument, *argument rebuttals* or *counter-arguments* that could be *defeaters* for $\langle \mathcal{A}, L \rangle$ are considered, *i.e.*, counter-arguments that by some criterion are preferred to $\langle \mathcal{A}, L \rangle$. For example, $\mathcal{A}_1$ is a counter-argument for $\mathcal{A}_2$, and *vice versa*. Since counter-arguments are arguments, defeaters for them may exist, and defeaters for these defeaters, and so on. Thus, a sequence of arguments called *argumentation line* is constructed, where each argument defeats its predecessor in the line (for a detailed explanation of this dialectical process see [10]). The prevailing argument provides a warrant for the information it supports. In DeLP, a literal L is *warranted* from (Π, Δ) if a non-defeated argument $\mathcal{A}$ supporting L exists. For instance, $\mathcal{A}_1$ and $\mathcal{A}_2$ are two conflicting arguments and, as will be explained next, $\mathcal{A}_1$ is a proper defeater for $\mathcal{A}_2$, thus warranting the literal $choose(khep1, box2)$.

In DeLP, the comparison criterion among arguments can be established in a modular way. Thus, an appropriate domain-dependent criterion can be used. The criterion that we will use in our approach is defined next.

Definition 1 (L-order). *Let $\mathcal{P}$ be a DeLP program and lits($\mathcal{P}$), the set of literals in $\mathcal{P}$. An* L-order *over $\mathcal{P}$ is a partial order over the elements of lits($\mathcal{P}$).*

Regarding physical aspects of the robots, their autonomy is limited, and they cannot measure the state of their batteries. Because of this drawback, a *greedy* strategy is used to select the next box. Therefore, the robot will *prefer its nearer boxes, then the boxes nearer to the store, and finally the smaller ones.* These preferences will be explicitly established using a preference order among the literals *smaller*, *nearer_store*, *nearer_robot* and *nearest_robot*. As will be shown below, an L-order must be provided as a set of facts within the program. These facts are written as $X > Y$, stating that a literal X is preferred to a literal Y, and they will be used to decide when an argument is better than another. In particular, the L-order defined in Fig. 3 represents the robots preferences over boxes, *i.e.*, it shows the L-order defined over program $\mathcal{P}_k$ (Fig. 2), used in our application examples. Based on a given L-order, the following argument comparison criterion can be defined.

Definition 2. *Let $\mathcal{P} = (\Pi, \Delta)$ be a DeLP-program and let "$<$" be an L-order over $\mathcal{P}$. Given two argument structures $\langle \mathcal{A}_1, h_1 \rangle$ and $\langle \mathcal{A}_2, h_2 \rangle$, the argument $\langle \mathcal{A}_1, h_1 \rangle$ will be preferred over $\langle \mathcal{A}_2, h_2 \rangle$ iff:*
1. there are two literals L_1 and L_2 such that $L_1 \in^* \mathcal{A}_1$, $L_2 \in^* \mathcal{A}_2$, $L_2 < L_1$, and
2. there are no literals L_1' and L_2' such that $L_1' \in^* \mathcal{A}_1$, $L_2' \in^* \mathcal{A}_2$, and $L_1' < L_2'$.
where $L \in^* \mathcal{A}$ iff there exists a defeasible rule $(L_0 \prec L_1, L_2, \ldots, L_n)$ in $\mathcal{A}$ and $L = L_i$ for some i $(0 \leq i \leq n)$.

On the basis of this criterion, defeaters can be either proper or blocking. Given an argument comparison criterion C, an argument $\mathcal{A}_1$, and a counter-argument $\mathcal{A}_2$ for it: $\mathcal{A}_2$ is a proper defeater for $\mathcal{A}_1$ iff $\mathcal{A}_2$ is better than $\mathcal{A}_1$ wrt. C; $\mathcal{A}_2$ is a blocking defeater for $\mathcal{A}_1$ iff neither of both is better than the other wrt. C; finally, $\mathcal{A}_2$ is not a defeater for $\mathcal{A}_1$ iff $\mathcal{A}_1$ is better than $\mathcal{A}_2$ wrt. C.

In DeLP, a query Q has four possible answers: YES, if Q is warranted; NO, if the complement of Q is warranted; UNDECIDED, if neither Q nor its complement are warranted; and UNKNOWN, if Q is not in the signature of the program.

$nearest\_robot(R,Z) > nearer\_robot(R,X,Y)$	(19)	$nearer\_store(Z,X) > smaller(X,Y)$	(24)
$nearer\_robot(R,Z,X) > nearer\_store(X,Y)$	(20)	$nearer\_robot(R,Z,X) > smaller(X,Y)$	(25)
$nearer\_robot(R,Z,Y) > smaller(X,Z)$	(21)	$nearer\_store(Z,X) > smaller(Y,Z)$	(26)
$nearer\_robot(R,Z,Y) > nearer\_store(X,Z)$	(22)	$nearest\_robot(R,Z) > smaller(X,Y)$	(27)
$nearest\_robot(R,Z) > nearer\_store(X,Y)$	(23)	$smaller(Z,X) > smaller(X,Y)$	(28)

Fig. 3. L-order over lits($\mathcal{P}_k$)

3 Selecting Boxes through Argumentation

In classical decision making domains it is usually assumed that the agent's choice behavior is modeled with a binary *preference relation* $\succsim$ where, $x \succsim y$ means that "x is at least as good as y". From $\succsim$ we can derive the *strict preference* relation

$\succ$[4] and the *indifference* relation $\sim$.[5] It is common to require the preference relation $\succsim$ to be *rational* (complete and transitive) and this is a necessary condition if $\succsim$ will be represented by a utility function. According to this *preference-based approach* (PBA), we should prove that the DeLP-program of Fig. 2 and the argumentation process involved when comparing two boxes, implement an implicit rational preference relation between any pair of alternatives (boxes) that could be presented to the robot. Despite of the relevance of the PBA from a theoretical point of view, an analysis of this type is beyond the scope of this paper. Besides, it is not always guaranteed that this approach accurately reflects the decision making capabilities of agents facing decision problems of the real world.[6]

In our work, we attempt to show how flexible can be an approach based on defeasible argumentation when new and changing information has to be considered during the decision making process. For this reason, the *static* properties of the preferences of our robots are not so important, and it should be useful to directly consider the *choice behavior* of the robots and to analyze the *dynamics* of the decision processes, when new alternatives have to be considered due to changes in the environment or when new information is available to the robot. From this point of view, an interesting and more flexible formal model of theory of decision making, called *choice-based approach* [11] provides us more adequate tools for evaluating the dynamics involved in the decisions of the robot.

The *choice-based approach* (CBA) takes the individual's choice behavior as a primitive object, which is represented by means of a *choice structure* $(\mathcal{B}, C(\cdot))$ where $\mathcal{B}$ is a family (a set) of subsets of X (the set of possible alternatives). Intuitively, each set $B \in \mathcal{B}$ (where $B \subset X$) represents each set of alternatives (or *choice experiments*) that can be conceivably posed to the decision maker. In this way, if $X = \{x, y, z\}$ and $\mathcal{B} = \{\{x, y\}, \{x, y, z\}\}$, we will assume that the sets $\{x, y\}$ and $\{x, y, z\}$ are valid choice experiments to be presented to the decision maker. $C(\cdot)$ is a *choice rule* (a correspondence) which basically assigns to each set of alternatives $B \in \mathcal{B}$ a non-empty set that represents the alternatives that the decision maker *might* choose when presented the alternatives in B. We can note that $C(B) \subset B$ for every $B \in \mathcal{B}$ and when $C(B)$ contains a single element, this element represents the *individual's choice* from among the alternatives in B. The $C(B)$ set may, however, contain more than one element and in this case they represent the *acceptable alternatives* in B for the agent.

A central assumption in this approach, the *weak axiom of revealed preference* (WARP), imposes an element of consistency on choice behavior in a sense paralleling the rationality assumptions of the PBA. Intuitively, the WARP principle reflects the expectation that an individual's observed choices will display a certain amount of consistency. For example, if an individual chooses alternative x (and only that) when facing the alternatives $\{x, y\}$, we would be surprised to

[4] Defined as $x \succ y \Leftrightarrow x \succsim y$ but not $y \succsim x$ and read "x is preferred to y"

[5] Defined as $x \sim y \Leftrightarrow x \succsim y$ and $y \succsim x$ and read "x is indifferent to y"

[6] It is well known in the decision theory community that completeness and transitivity assumptions are usually very hard to satisfy in real world agents when evaluating alternatives far from common experience [11].

see him choose y when faced the alternatives $\{x, y, z\}$. More formally, the weak axiom postulates that "if there is some choice experiment $B \in \mathcal{B}$ such that x and y are presented as alternatives $(x, y \in B)$ and "x is *revealed at least as good as* y $(x \in C(B))$ then it does not exist other choice experiment $B' \in \mathcal{B}$ where "y is *revealed preferred* to x $(x, y \in B', y \in C(B')$ and $x \notin C(B'))$.

In our work, the robots are essentially facing a discrete multi-attribute decision problem, where each alternative (box) would represent a complete assignment of attribute values. The set X of alternatives available to the robots would be represented by the properties associated to the boxes to be considered in the decision process. For this reason, we will use the notation x_{box_i} to denote the description of the attributes corresponding to box box_i, *e.g.*, "near to the robot," "near to the store," etc.

It is direct to note the similarities of our approach with the scenario assumed in the CBA; where the *choose*/2 predicate (Fig. 2) would play the role of the choice rule $C(\cdot)$. In this case, if n boxes $box_1, \ldots, box_n$ are present in the environment, the robot faces a choice experiment with n alternatives $\{x_{box_1}, \ldots, x_{box_n}\}$ with the properties corresponding to each box present in the environment. In this context, when the DeLP answer to the query $choose(R, box_k)$ is YES, it should be interpreted as saying that the properties of box_k make it an acceptable alternative for the robot R, or more formally, the robot R has a choice rule $C(\cdot)$ such that $x_{box_j} \in C(\{x_{box_1}, \ldots, x_{box_n}\})$. From this perspective, is direct that the WARP principle can be easily restated in terms of the *choose* predicate. We can say in this case, that *choose*/2 satisfies the weak axiom of revealed preference if every time that two boxes with attributes x_j and x_k are considered and *choose* answers YES to the box with attributes x_j, it does not exist another situation where two boxes with the same attributes are considered and the answer of *choose* for x_k is YES and it does not respond YES for the box x_j.

Below, we show through several examples how the robots react in different situations that might arise in our dynamic application domain. Besides, it will be pointed out how the robots decisions respect the WARP principle.

Example 1. Let us consider first a simple situation (Fig. 1(a)) with one robot (*khep*1) and three boxes: *box*1 (small), *box*2 (big) and *box*3 (medium). Here, *box*2 and *box*3 are near to the store, *box*2 is near to the robot, and *box*1 is far from both, robot and store. Considering *khep*1*'s* preferences (Fig. 3) *box*2 should be chosen because, despite of its size, is the only box near to the robot and also to the store. The robot's perception with regards to this situation is the set:

$$\Pi_a = \left\{ \begin{array}{l} smaller(box1, box2),\ nearer\_store(box2, box1),\ nearer\_robot(khep1, box2, box3), \\ smaller(box1, box3),\ nearer\_store(box3, box1),\ nearer\_robot(khep1, box2, box1), \\ smaller(box3, box2) \end{array} \right\}$$

The DeLP-program of *khep*1 includes $\mathcal{P}_k$, Π_a and the L-order of Fig. 3. From this program, DeLP builds the following undefeated arguments:

$$\mathcal{A}_{1.1} = \left\{ \begin{array}{l} \sim choose(khep1, box1) \prec better(khep1, box2, box1) \\ better(khep1, box2, box1) \prec nearer\_store(box2, box1) \end{array} \right\}$$

$$\mathcal{A}_{1.2} = \left\{ \begin{array}{l} choose(khep1, box2) \prec better(khep1, box2, box1) \\ better(khep1, box2, box1) \prec nearer\_store(box2, box1) \end{array} \right\}$$

$$\mathcal{A}_{1..3} = \left\{ \begin{array}{l} \sim choose(khep1, box3) \prec better(khep1, box2, box3) \\ better(khep1, box2, box3) \prec nearer\_robot(khep1, box2, box3) \end{array} \right\}$$

Hence, as expected from the preferences encoded in Fig. 3, the DeLP answer for $choose(khep1, box1)$ is NO, for $choose(khep1, box2)$ is YES, and for $choose(khep1, box3)$ is NO.

Example 2. In order to illustrate how changes in a dynamic environment are handled by the DeLP-program $\mathcal{P}_k$, consider that in the scenario depicted in Fig. 1(a) a new medium size box ($box4$) is placed on top of $box2$. It is clear that $box2$ is no longer the best choice, and the robot should select $box4$. In this new scenario, the fact $on(box4, box2)$ is added, and rule (17) blocks any argument supporting $choose(khep1, box2)$. Hence, the DeLP answer for $choose(khep1, box2)$ is NO, whereas $choose(khep1, box4)$ returns YES.

Example 3. Consider the situation of Fig. 1(b), with one robot ($khep1$) and four boxes: $box1$ (small), $box2$ (small), $box3$ (medium) and $box4$ (big). Here, $box1$ and $box2$ are near to the robot and to the store, $box3$ is far from the robot and near to the store, and $box4$ is far from both, robot and store. The robot's perception is:

$$\Pi_b = \left\{ \begin{array}{ll} smaller(box1, box3), & smaller(box1, box4), \\ smaller(box2, box3), & smaller(box2, box4), \\ smaller(box3, box4), & nearer\_store(box1, box4), \\ nearer\_store(box2, box4), & nearer\_store(box3, box4), \\ nearest\_robot(khep1, box1), & nearest\_robot(khep1, box2), \\ nearer\_robot(khep1, box1, box3), & nearer\_robot(khep1, box2, box3), \\ nearer\_robot(khep1, box1, box4), & nearer\_robot(khep1, box2, box4), \\ same\_properties(khep1, box1, box2) & \end{array} \right\}$$

It is clear that $box1$ and $box2$ are better alternatives than $box3$ and $box4$, hence, the latter ones should not be chosen. Nonetheless, as $box1$ and $box2$ have the same properties (from $khep1$'s point of view) these alternatives should be indifferent to $khep1$. The DeLP-program of $khep1$ will include $\mathcal{P}_k$, Π_b and the L-order of Fig. 3. From this program, the answer for both $choose(khep1, box1)$ and $choose(khep1, box2)$ will be YES, representing that the robot presents equal preference (*i.e.*, indifference) to both boxes. As it was expected, the answers for $choose(khep1, box3)$ and $choose(khep1, box4)$ will be NO.

Example 4. Taking into account the WARP principle mentioned at the beginning of this section, the choices made by $khep1$ in Ex. 3 should not be changed if $box3$ and $box4$ were removed from the scenario. This is because these boxes (alternatives) are worse than $box1$ and $box2$, which have the same properties independently of the presence or absence of other boxes. If we consider a scenario that only includes $box1$ and $box2$, the robot's perception would be:

$$\Pi_{b'} = \left\{ \begin{array}{ll} nearest\_robot(khep1, box1), & nearest\_robot(khep1, box2), \\ same\_properties(khep1, box1, box2) & \end{array} \right\}$$

From the DeLP-program including $\mathcal{P}_K$, $\Pi_{b'}$ and the L-order of Fig. 3, as expected, the answer to $choose(khep1, box1)$ and $choose(khep1, box2)$ is YES.

Example 5. As shown in Ex. 4, some decisions should be maintained if the changes in the world are not significant to the robot's decision-making policy.

Nonetheless, if we consider the situation shown in Fig. 1(c), where a new robot ($khep2$) appears in the environment, this fact might modify $khep1's$ choice depending on $khep2's$ position. Now, $box1$ is the nearest box of $khep2$ and from its point of view $box1$ is its best choice. In face of this new evidence, $khep1$ still have a nearest box ($box2$) to choose instead of $box1$. Since the only difference from the scenario depicted in Fig. 1(b) is the presence of $khep2$, the perception now is $\Pi_c = \Pi_b \cup \Phi$, where:

$$\Phi = \left\{ \begin{array}{ll} nearest\_robot(khep2, box1), & nearer\_robot(khep2, box1, box2), \\ nearer\_robot(khep2, box1, box3), & nearer\_robot(khep2, box1, box4) \end{array} \right\}$$

At first sight, the alternatives (boxes) available to $khep1$ seem to be the same that in Ex. 3. Nevertheless, with the presence of $khep2$ in the environment, the alternative associated with $box1$ changes because, as represented in Φ, $box1$ has the new attribute of being the nearest box of another robot ($khep2$).

Note that rule (15) plays a fundamental role, because it allows the robots to choose a certain box, considering the other robots' preferences. Without this rule $khep1$ would have chosen $box1$ and $box2$ as in Ex. 3. Then, a problematic situation can arise if both robots intend to grab the same box.

Since both robots have the same preferences and the same perception (Π_c), their DeLP-program will include $\mathcal{P}_k$, Π_c and the L-order of Fig. 3. From this program, the answers for $choose(khep1, box3)$ and $choose(khep1, box4)$ remains being NO, whereas the answer for $choose(khep1, box2)$ remains being YES, since the presence of $khep2$ does not compromise $khep1$'s decision. In opposition, the query $choose(khep1, box1)$ is now UNDECIDED, because it is a box that could be selected by $khep2$. Regarding $khep2$, the DeLP answer for $choose(khep2, box1)$ will be YES, and for $choose(khep2, box2)$, $choose(khep2, box3)$ and $choose(khep2, box4)$ will be NO, since they do not represent good choices for this robot.

Example 6. Let us consider a final example regarding the situation depicted in Fig. 1(d). As can be observed, there are three small boxes ($box1$, $box2$ and $box3$) and two robots ($khep1$ and $khep2$). All the boxes are near to the store and in particular $box1$ is near to both robots. Then, $box2$ is near to $khep1$ and far from $khep2$ while $box3$ is near to $khep2$ and far from $khep1$. The boxes $box1$ and $box2$ have the same properties from $khep1's$ standpoint, the same occurs with $box1$ and $box3$ from $khep2's$ point of view. Since both robots can choose other boxes besides $box1$, and they are as good as $box1$, intuitively the best combination would be $khep1$ choosing $box2$ with $khep2$ choosing $box3$. The robots' perception in this situation is the following set:

$$\Pi_d = \left\{ \begin{array}{ll} nearest\_robot(khep1, box1), & nearest\_robot(khep1, box2), \\ nearest\_robot(khep2, box1), & nearest\_robot(khep2, box3), \\ nearer\_robot(khep1, box2, box3), & nearer\_robot(khep2, box3, box2), \\ same\_properties(khep1, box1, box2), & same\_properties(khep2, box1, box3) \end{array} \right\}$$

From the DeLP-program that includes $\mathcal{P}_k$, Π_d and the L-order of Fig. 3, as expected, the answers for $choose(khep1, box2)$ and $choose(khep2, box3)$ are YES, whereas the answer for $choose(khep1, box3)$ and $choose(khep2, box2)$ are NO. It is important to note that the answer for both queries $choose(khep1, box1)$ and $choose(khep2, box1)$ is UNDECIDED. The reason why this occurs is that DeLP cannot build an undefeated argument to support neither $choose(khep1, box1)$ nor $\sim choose(khep1, box1)$, and the same situation holds for $khep2$.

Changing Preferences

To sum up, it worths noticing that this defeasible approach to decision making is flexible enough to easily modify preferences, and thus obtain a different behavior from the robot. As stated in Sect. 2, in the above examples a robot prefers its nearer boxes, then the boxes nearer to the store, and finally the smaller ones. These preferences were explicitly established by the L-order given in Fig. 3, but they can be replaced in a modular way. For instance, another possibility would be giving highest preference to the smallest boxes, *i.e.*, the size of each box would be more important than its proximity to the store or to the robot. This would imply to change in Fig. 3 the facts (21), (24), (25), and (26) for the following ones: $smaller(Z, Y) > nearer\_store(X, Z)$, $smaller(Z, X) > nearer\_robot(R, Y, Z)$ and $smaller(X, Y) > nearest\_robot(R, Y)$. Taking into account these changes, let us reconsider Ex. 1. Now, the DeLP answers to the queries $choose(khep1, box1)$, $choose(khep1, box2)$, and $choose(khep1, box3)$ will be YES, NO, and NO, respectively. Thus, according to this new L-order, $khep1$ now chooses $box1$ (the smallest one) instead of $box2$ (the nearest one).

4 Conclusions and Related Work

In this paper we have shown how a Defeasible Logic Programming approach could be applied in a robotic domain for knowledge representation and reasoning about which task to perform next. Our approach considers the ability of Defeasible Logic Programming to reason with incomplete and potentially inconsistent information. Through several examples we aimed to show the flexibility of this approach to program the robots' preference policy, by considering a simple application domain where real *Khepera* 2 robots perform cleaning tasks.

In particular, in our work we follow some ideas exposed by Parsons *et al.* [12] about the integration of high-level reasoning facilities with low-level robust robot control. We share the approach of seeing the low-level module as a black box which receives goals to be achieved from the high-level component, and plans to reach goals are internally generated. However, our work differs from [12] in that we do not use a BDI deliberator as high-level reasoning layer, instead we use a non-monotonic reasoning module based on a defeasible argumentation system.

With respect to this last issue, our approach to decision making is related to other works which use argumentative processes as a fundamental component in the decision making of an agent [13–15]. It is important to note that these argumentation systems have been usually integrated in *software* agents. On the other hand, in our approach, defeasible argumentation is applied in a robotic domain where uncertainty (generated by noisy sensors and effectors), changes in the physical environment, and incomplete information about it, make this kind of problems a more challenging test-bed for the decision processes of an agent.

Finally, since Prolog is a programming language that has already been used to develop applications in the field of cognitive robotics [16], we will make some final brief comments on the differences that arise when programming a cognitive agent in Prolog with respect to DeLP. A comparison of this nature can be rather unfair, since DeLP was intended from the beginning to be a high-level

logic programming language for knowledge representation and reasoning. In our approach, robots' preferences are modeled as a partial order among literals. Therefore, in DeLP, when these preferences are changed, it will only affect the set of facts representing them. That is because the DeLP formalism allows for a more modular way to represent knowledge. In opposition to this, in Prolog, changing the preferences may involve checking and rewriting most of the program. Hence, the developer would have to specify each and every scenario within which a robot makes a certain choice.

Acknowledgment

We thank the Universidad Nacional de San Luis and the Universidad Nacional del Sur for their unstinting support. This work is partially supported by, CONICET (PIP 5050), and ANPCyT (PICT 2002, Nro.13096 and Nro.12600).

References

1. Jeffrey, R.C.: The Logic of Decision. 2nd edn. University Of Chicago Press (1990)
2. Doyle, J., Thomason, R.H.: Background to qualitative decision theory. AI Magazine (1999)
3. Rao, A., Georgeff, M.: Decision procedures for BDI logics. Journal of Logic and Computation (1998)
4. Brooks, R.A.: A robust layered control system for a mobile robot. IEEE Journal of Robotics and Automation (1986)
5. K-Team: Khepera 2. http://www.k-team.com (2002)
6. Gat, E.: On three-layer architectures. In: Artificial Intelligence and Mobile Robots. (1998)
7. Estlin, T., Volpe, R., Nesnas, I., Muts, D., Fisher, F., Engelhardt, B., Chien, S.: Decision-making in a robotic architecture for autonomy. In: International Symposium, on AI, Robotics and Automation for Space. (2001)
8. Rotstein, N., García, A., Simari, G.: Reasoning from desires to intentions: A dialectical framework. In: Proceedings of the 22nd. AAAI Conference on Artificial Intelligence. (2007) 136–141
9. Ferretti, E., Errecalde, M., García, A., Simari, G.: KheDeLP: A framework to support defeasible logic programming for the khepera robots. In: ISRA. (2006)
10. García, A., Simari, G.: Defeasible logic programming: An argumentative approach. Theory Practice of Logic Programming 4(1) (2004) 95–138
11. Mas-Collel, A., Whinston, M.D., Green, J.R.: Microeconomic Theory. Oxford University Press (1995)
12. Parsons, S., Pettersson, O., Saffiotti, A., Wooldridge, M.: Robots with the Best of Intentions. In: Artificial Intelligence Today: Recent Trends and Developments. Springer (1999)
13. Atkinson, K., Bench-Capon, T.J.M., Modgil, S.: Argumentation for decision support. In: DEXA. (2006) 822–831
14. Kakas, A., Moraitis, P.: Argumentation based decision making for autonomous agents. In: AAMAS. (2003)
15. Parsons, S., Fox, J.: Argumentation and decision making: A position paper. In: FAPR. (1996) .
16. Levesque, H.J., Pagnucco, M.: Legolog: Inexpensive experiments in cognitive robotics. In: 2nd International Cognitive Robotics Workshop. (2000)