Defeasible Decision Making in a Robotic Environment

Edgardo Ferretti, Marcelo Errecalde
Laboratorio de Investigación y Desarrollo en Inteligencia Computacional (LIDIC)*
Departamento de Informática, Universidad Nacional de San Luis
Ejército de los Andes 950 - Local 106
(D5700HHW) - San Luis - Argentina
Tel: (02652) 420823 / Fax: (02652) 430224
e-mail: {ferretti, merreca}@unsl.edu.ar

and

Alejandro García,‡ Guillermo Simari
‡ Consejo Nacional de Investigaciones Científicas y Técnicas
Laboratorio de Investigación y Desarrollo en Inteligencia Artificial (LIDIA)†
Departamento de Ciencias e Ingeniería de la Computación, Universidad Nacional del Sur
Av. Alem 1253, (B8000CPB) Bahía Blanca, Argentina
Tel: (0291) 459-5135 / Fax: (0291) 459-5136
e-mail: {ajg, grs}@cs.uns.edu.ar

Abstract

Decision making models for autonomous agents are recently receiving increased attention, particularly in the field of intelligent robots. This work presents a Defeasible Logic Programming approach to decision making in an environment with single and multiple robots. We will show, 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, we will explain with detailed examples how the decision process is performed when there is only one robot in the environment, and then we will consider how the same robot decides when there are more robots working in the environment.

Keywords: Cognitive Robotics, Decision making, DeLP, Khepera robot, Webots.

Resumen

Actualmente, los modelos de toma de decisiones para agentes autónomos están recibiendo mucha atención, particularmente en el área de robots inteligentes. Este trabajo presenta un enfoque basado en Programación en Lógica Rebatible para la toma de decisiones en un ambiente con un único robot y con múltiples robots. Mostraremos como una herramienta exitosa para la representación de conocimiento y razonamiento rebatible, puede ser aplicada al problema de decidir que tarea debe ser realizada a continuación. Además, explicaremos con ejemplos detallados como se realiza el proceso de decisión cuando hay solamente un robot en el ambiente, y luego consideraremos como decide el mismo robot cuando hay otros robots presentes en el ambiente.

Palabras claves: Robótica Cognitiva, Toma de decisiones, DeLP, robot Khepera, Webots.

*Partially supported by Universidad Nacional de San Luis and the ANPCyT (PICT 2002, Nro. 12600).
†Partially supported by Universidad Nacional del Sur, the ANPCyT (PICT 2002, Nro. 13096) and CONICET (PIP 5050).

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 [13], Qualitative Decision Theory [6] and BDI logics [22]. 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 [5]. An established approach to decision making in robotic systems is that of reactive decision systems. In such systems it is assumed that the agent's success is determined by its capacity to appropriately react solely to external stimuli. The decision process is thus 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 this approach to decision making becomes 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? Which one of the alternative plans do I have to select to reach a certain goal? Can I carry out this task on my own or I should request help from another agent?* Further complicating matters, the information used in the decision process is (in most domains) incomplete and potentially inconsistent.

To address these issues, we propose the use of a defeasible argumentation formalism. 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, consisting of different scenarios where simulated robots perform cleaning tasks. We use the professional simulator (see Figure 1) *Webots* [18], to simulate the *Khepera* 2 robots [14]. The *Khepera* 2 robot is a miniature mobile robot which has eight infrared sensors to measure ambient light levels and proximity to nearby objects. Two independent variable speed motors enable forward and backward motion as well as turns at different speeds. A gripper-arm extension module allows the handling of small objects.

The environment (see Figure 1) 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 (Figure 1(c)), 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.

Due to physical constraints on the capabilities of the gripper-arms, the robots can grab small and medium boxes, but because of their size the big ones cannot be handled. Nevertheless, a robot is able to move a big box by pushing it. At most two boxes can be stacked, but a box cannot be stacked on top of a smaller box. Therefore, big boxes are always on the floor.

The autonomy of the robots is limited and they cannot measure the state of their batteries, thus, the robots could not perform a globally optimized task. Because of this drawback, a *greedy* strategy is used to select the next box. 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 [12, 7, 23]. In this work, we only consider the necessary reasoning processes to make decisions about which is the most suitable box to be transported by the robots. We will not address low-level aspects related to sensorial perception, like image acquisition and processing. Besides, we are not going to consider problems related to the implementation of low-level actions, like object handling and the robots' navigation system. Some of the aspects related to the sensorial and effectorial support for the Khepera robots have been presented elsewhere [8, 9].

The paper is organized as follows. In Section 2 we explain how robots represent their knowledge and include an overview of the reasoning formalism they use. Section 3 shows how the decision process is performed when there is only one robot in the environment. Then, in Section 4 we show how the robot decides when there are more robots working in the environment. In Section 5 related work is described. Finally, Section 6 offers some conclusions.

(a) (b) (c)

Figure 1: Three different scenarios

2 KNOWLEDGE REPRESENTATION AND REASONING

Figure 1(b) shows an example where there is only one robot ($khep1$) and four boxes: two small ($box1$ and $box2$) which are at the same distance from the robot, $box3$ that is medium size and is near to the store, and $box4$ that is big and is far from both, robot and store. Considering its preferences the robot will obtain arguments for and against selecting each box.

For example, there is an argument *for* selecting $box1$ because is near to the robot and is small, but there is an argument *against* selecting $box1$ (counter-argument) because there is another small box near to the robot ($box2$) that is nearer to the store than $box1$. As will be shown below a dialectical analysis involving arguments and counter-arguments will be performed to decide which argument prevails. In this case the box chosen will be $box2$, because it is small, and is nearer to the store. Since the environment is dynamic, if something changes, then, new arguments can be generated or other can be invalidated. Thus, the selected box may be different. For instance, let us consider Figure 1(c), that differs from Figure 1(b) in that there is

one more robot ($khep2$) in the environment. Here, using this new information, the robot $khep1$ will choose $box1$ because it has a new argument against selecting $box2$:"there is another robot ($khep2$) that will choose $box2$ on the grounds that it is its nearest smaller box." In this way, the overall performance is enhanced avoiding a conflict in the robots' selection choices.

The robot's knowledge about the environment and its preferences for selecting a box will be represented using Defeasible Logic Programming (DeLP) [10] a formalism that combines logic programming and defeasible argumentation.

In DeLP, knowledge is represented with a program $\mathcal{P}$ that contains facts, strict rules and defeasible rules. *Facts* are ground literals representing atomic information or the negation of atomic information. In our application examples, facts will be used for representing perceptual information about the environment, (*e.g.*, $box(box2)$ or $near(box2, store)$). *Strict Rules*, are denoted $L_0 \leftarrow L_1, \ldots, L_n$, where the *head* L_0 is a ground literal and the *body* $\{L_i\}_{i>0}$ is a set of ground literals. These kind of rules will be used for representing firm information (*e.g.*, $\sim near(box1, khep1) \leftarrow far(box1, khep1)$). On the other hand, *Defeasible Rules*, denoted $L_0 \prec L_1, \ldots, L_n$, where the *head* L_0 is a ground literal and the *body* $\{L_i\}_{i>0}$ is a set of ground literals, represent tentative reasons for (or against) selecting a box (*e.g.*, $choose(X) \prec small(X)$, or $\sim choose(X) \prec small(X), far(X, khep1)$).

Syntactically, the symbol "$\prec$" is all that distinguishes a defeasible rule from a strict one. Pragmatically, a defeasible rule is used to represent defeasible knowledge, *i.e.*, tentative information that may be used if nothing could be posed against it. A defeasible rule "$Head \prec Body$" is understood as expressing that "*reasons to believe in the antecedent Body provide reasons to believe in the consequent Head*" [24]. When required, a Defeasible Logic Program $\mathcal{P}$ is denoted (Π, Δ) where $\Pi{=}\Pi_f \cup \Pi_r$, distinguishing the subset Π_f of facts, strict rules Π_r, and the subset Δ of defeasible rules. Observe that strict and defeasible rules are ground. However, following the usual convention [16], some examples will use "schematic rules" with variables. Given a "schematic rule" R, $Ground(R)$ stands for the set of all ground instances of R. Given a program $\mathcal{P}$ with schematic rules, we define: $Ground(\mathcal{P}) = \bigcup_{R \in \mathcal{P}} Ground(R)$. In order to distinguish variables, they are denoted with an initial uppercase letter.

Strong negation is allowed in the head of program rules, and hence may be used to represent contradictory knowledge. From a program (Π, Δ) contradictory literals could be derived, however, the set Π (which is used to represent non-defeasible information) must possess certain internal coherence. Therefore, Π has to be non-contradictory, *i.e.*, no pair of contradictory literals can be derived from Π. Given a literal L the complement with respect to strong negation will be denoted $\overline{L}$ (*i.e.*, $\overline{a}{=}\sim a$ and $\overline{\sim a}{=}a$.)

To deal with contradictory and dynamic information, in DeLP, *arguments* for conflicting pieces of information are built and then compared in order to decide which one prevails. The argument that prevails provides a *warrant* for the information that it supports. A brief explanation of how warrants are obtained using DeLP is included below (the interested reader is referred to [10] for a detailed explanation.[1])

In DeLP a literal L is *warranted* from (Π, Δ) if a non-defeated argument $\mathcal{A}$ supporting L exists. To put it briefly, 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 $\mathcal{A} \cup \Pi$. To establish if $\langle \mathcal{A}, L \rangle$ is non-defeated, *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 $\langle \mathcal{A}, L \rangle$. Since counter-arguments are arguments, defeaters for them may exist, and defeaters

[1] The implementation (interpreter) of DeLP that satisfies the semantics described in [10] is currently accessible online at `http://lidia.cs.uns.edu.ar/DeLP`.

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].) In DeLP, given a query Q there are 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 language of the program.

3 SINGLE ROBOT BOX SELECTION

In this section we describe the processes involved in deciding which box to transport when there is only one robot in the environment.

Example 1 *Consider the simple scenario depicted in Figure 1(a) where there is a single robot (khep1), and three boxes: box1 (small) and box4 (big) near to the robot, and box3 (medium) near to the store. The knowledge of the robot, referring to this particular scenario, will be represented with the defeasible logic program* $\mathcal{P}_1 = (\Pi_1, \Delta_1)$ *shown in Figures 2 and 3.*

The set Π_1 contains a subset Π_{f1} of facts representing the perception of the current situation (as shown in Figure 2(a)) and a subset Π_{r1} of strict rules shown in Figure 2(b). The facts of Π_{f1} are obtained from perception functions (see [8]) and represent information about the objects present in the environment (facts (1)-(8)) and the position of the objects (facts (9)-(15)). The strict rules in Figure 2(b) represent non-defeasible information, for example, rule (16) states that if X is far from an object O (in this case O can be instantiated with, $khep1$ or $store$) then it is not near to O. Rules (17)-(24) define size relationships among the boxes. In rule (25) is stated that an object X is nearer than an object Y with respect to an object O, if X is near to O and Y is not. Besides, rules (26) and (27) define strict reasons for choosing a box X. For instance, if X is the last box in the environment this is a firm reason to be chosen (rule (26)) but if X has a box on its top, this is a negative reason for choosing it (rule (27)).

$robot(khep1)$	(1)	$on(box1, box4)$	(9)	$\sim near(X,O) \leftarrow far(X,O)$	(16)
$self(khep1)$	(2)	$near(box1, khep1)$	(10)	$same\_size(X,Y) \leftarrow small(X), small(Y)$	(17)
$box(box1)$	(3)	$near(box4, khep1)$	(11)	$same\_size(X,Y) \leftarrow medium(X), medium(Y)$	(18)
$box(box3)$	(4)	$near(box3, store)$	(12)	$same\_size(X,Y) \leftarrow big(X), big(Y)$	(19)
$box(box4)$	(5)	$far(box1, store)$	(13)	$smaller(X,Y) \leftarrow small(X), medium(Y)$	(20)
$small(box1)$	(6)	$far(box3, khep1)$	(14)	$smaller(X,Y) \leftarrow small(X), big(Y)$	(21)
$medium(box3)$	(7)	$far(box4, store)$	(15)	$smaller(X,Y) \leftarrow medium(X), big(Y)$	(22)
$big(box4)$	(8)			$\sim smaller(X,Y) \leftarrow same\_size(X,Y)$	(23)
				$\sim smaller(X,Y) \leftarrow smaller(Y,X)$	(24)
				$nearer\_than(X,Y,O) \leftarrow near(X,O), \sim near(Y,O)$	(25)
				$choose(X) \leftarrow unique(X)$	(26)
				$\sim choose(X) \leftarrow on(Y,X)$	(27)

(a) Π_{f1} (b) Π_{r1}

Figure 2: Facts and strict rules of Π_1

Figure 3 includes the defeasible rules of Δ_1. Rules (28)-(30) provide defeasible reasons to determine if a box X is smaller than a box Y when strict rules (17)-(24) cannot be used, because of the lack of information about the boxes' size. In the same way, rules (31)-(35) provide evidence to the determine if a box X is nearer to the robot (or to the store), than a box Y if there is no enough information about the proximity of these boxes to the robot (or to the store). If enough information is available the strict rule (25) can be used instead. Furthermore, rules (36)-(51) model preference criteria with respect to the size and location of the boxes. For example, rules (36) and (37) represent the preferences of the robot with respect

to the boxes' size, the smaller ones are preferred. Moreover, rules (40) and (44) show that being near to the store or to the robot has a higher priority than the boxes' size. In addition, rule (48) and (50) state that boxes near to the robot are more desirable than those near to the store, independently of their size. Rules (52) and (53) represent the (defeasible) criteria used by the robot to choose the box to be transported. These rules, differ from rules (26) and (27) in that they are based on the comparison between the features of two boxes. Thus, the rules to determine if the robot prefers a box X over a box Y ($pref(X,Y)$) play the role of a rational preference relation, as defined in classical decision theory [17]. The main difference of our approach lays in that the preferences are defeasible, they are based on an argumentation system that manages incomplete information about the environment, and in consequence they can change when new information is available.

$$\sim smaller(X,Y) \prec box(X), box(Y) \quad (28)$$
$$smaller(X,Y) \prec box(X), box(Y), small(X) \quad (29)$$
$$smaller(X,Y) \prec box(X), box(Y), big(Y) \quad (30)$$
$$\sim nearer\_than(X,Y,O) \prec box(X), box(Y) \quad (31)$$
$$nearer\_than(X,Y,O) \prec box(X), box(Y), near(X,O) \quad (32)$$
$$\sim nearer\_than(X,Y,O) \prec box(X), box(Y), \sim near(X,O) \quad (33)$$
$$\sim nearer\_than(X,Y,O) \prec box(X), box(Y), near(X,O), near(Y,O) \quad (34)$$
$$\sim nearer\_than(X,Y,O) \prec box(X), box(Y), \sim near(X,O), \sim near(Y,O) \quad (35)$$
$$pref(X,Y) \prec smaller(X,Y) \quad (36)$$
$$\sim pref(X,Y) \prec \sim smaller(X,Y) \quad (37)$$
$$pref(X,Y) \prec nearer\_than(X,Y,store), smaller(X,Y) \quad (38)$$
$$\sim pref(X,Y) \prec nearer\_than(Y,X,store), \sim smaller(X,Y) \quad (39)$$
$$pref(X,Y) \prec nearer\_than(X,Y,store), \sim smaller(X,Y) \quad (40)$$
$$\sim pref(X,Y) \prec nearer\_than(Y,X,store), smaller(X,Y) \quad (41)$$
$$pref(X,Y) \prec self(Z), nearer\_than(X,Y,Z), smaller(X,Y) \quad (42)$$
$$\sim pref(X,Y) \prec self(Z), nearer\_than(Y,X,Z), smaller(X,Y) \quad (43)$$
$$pref(X,Y) \prec self(Z), nearer\_than(X,Y,Z), \sim smaller(X,Y) \quad (44)$$
$$\sim pref(X,Y) \prec self(Z), nearer\_than(Y,X,Z), \sim smaller(X,Y) \quad (45)$$
$$pref(X,Y) \prec self(Z), nearer\_than(X,Y,Z), nearer\_than(X,Y,store), smaller(X,Y) \quad (46)$$
$$pref(X,Y) \prec self(Z), nearer\_than(X,Y,Z), nearer\_than(X,Y,store), \sim smaller(X,Y) \quad (47)$$
$$\sim pref(X,Y) \prec self(Z), nearer\_than(Y,X,Z), nearer\_than(X,Y,store), smaller(X,Y) \quad (48)$$
$$pref(X,Y) \prec self(Z), nearer\_than(X,Y,Z), nearer\_than(Y,X,store), smaller(X,Y) \quad (49)$$
$$\sim pref(X,Y) \prec self(Z), nearer\_than(Y,X,Z), nearer\_than(X,Y,store), \sim smaller(X,Y) \quad (50)$$
$$pref(X,Y) \prec self(Z), nearer\_than(X,Y,Z), nearer\_than(Y,X,store), \sim smaller(X,Y) \quad (51)$$
$$choose(X) \prec diff(X,Y), pref(X,Y) \quad (52)$$
$$\sim choose(X) \prec diff(X,Y), \sim pref(X,Y) \quad (53)$$

Figure 3: Defeasible rules of Δ_1

In the situation described in Example 1, it is clear that the robot should choose the small box ($box1$) because it is near to itself, it should not choose $box4$ because $box1$ is on its top and it should not choose $box3$ because there is a smaller box that can be chosen. From $\mathcal{P}_1$ there are four arguments supporting $choose(box1)$:

$$\mathcal{A}_1 = \left\{ \begin{array}{l} choose(box1) \prec diff(box1,box3), pref(box1,box3) \\ pref(box1,box3) \prec smaller(box1,box3) \end{array} \right\}$$

$$\mathcal{A}_2 = \left\{ \begin{array}{l} choose(box1) \prec diff(box1,box3), pref(box1,box3) \\ pref(box1,box3) \prec self(khep1), \\ \qquad nearer\_than(box1,box3,khep1), \\ \qquad smaller(box1,box3) \end{array} \right\}$$

$$\mathcal{A}_3 = \left\{ \begin{array}{l} choose(box1) \prec diff(box1,box3), pref(box1,box3) \\ pref(box1,box3) \prec self(khep1), \\ \qquad nearer\_than(box1,box3,khep1), \\ \qquad nearer\_than(box3,box1,store), \\ \qquad smaller(box1,box3) \end{array} \right\}$$

$$\mathcal{A}_4 = \left\{ \begin{array}{l} choose(box1) \prec diff(box1,box4), pref(box1,box4) \\ pref(box1,box4) \prec smaller(box1,box4) \end{array} \right\}$$

Since all the arguments $\mathcal{A}_1$-$\mathcal{A}_4$ have no defeaters, then the DeLP answer for $choose(box1)$ is YES. On the other hand, the answers for $choose(box3)$ and $choose(box4)$ are NO.

Example 2 *Consider now the situation presented at the beginning of this paper and depicted in Figure 1(b). There is a single robot (khep1) and two small boxes: box1 near to the robot, and*

box2 near to the robot and to the store. There are also a medium size box (box3) near to the store and a big one (box4) far from both, robot and store. The knowledge of khep1, referring to this particular scenario, will be represented with the defeasible logic program $\mathcal{P}_2 = (\Pi_2, \Delta_2)$*, where the perceptions of the current situation* (Π_{f2}) *are presented in Figure 4(a). Defeasible and strict rules of* $\mathcal{P}_2$ *coincide with the ones of* $\mathcal{P}_1$ *(Example 1), i.e.,* $\Pi_{r2}=\Pi_{r1}$*, and* $\Delta_2=\Delta_1$*.*

$robot(khep1)$	(54)	$big(box4)$	(63)
$self(khep1)$	(55)	$near(box1, khep1)$	(64)
$box(box1)$	(56)	$near(box2, khep1)$	(65)
$box(box2)$	(57)	$near(box2, store)$	(66)
$box(box3)$	(58)	$near(box3, store)$	(67)
$box(box4)$	(59)	$far(box1, store)$	(68)
$small(box1)$	(60)	$far(box3, khep1)$	(69)
$small(box2)$	(61)	$far(box4, khep1)$	(70)
$medium(box3)$	(62)	$far(box4, store)$	(71)

(a) From figure 1(b)

$robot(khep1)$	(72)	$near(box1, khep1)$	(83)
$robot(khep2)$	(73)	$near(box2, khep1)$	(84)
$self(khep1)$	(74)	$near(box2, khep2)$	(85)
$box(box1)$	(75)	$near(box2, store)$	(86)
$box(box2)$	(76)	$near(box3, store)$	(87)
$box(box3)$	(77)	$near(box4, khep2)$	(88)
$box(box4)$	(78)	$far(box1, store)$	(89)
$small(box1)$	(79)	$far(box1, khep2)$	(90)
$small(box2)$	(80)	$far(box3, khep1)$	(91)
$medium(box3)$	(81)	$far(box3, khep2)$	(92)
$big(box4)$	(82)	$far(box4, khep1)$	(93)
		$far(box4, store)$	(94)

(b) From figure 1(c)

Figure 4: $khep1's$ perceptual information

In the situation described in Example 2, it is evident that the robot should choose $box2$ because is near to itself and is also near to the store. From the program $\mathcal{P}_2$ there are seven arguments ($\mathcal{A}_5$-$\mathcal{A}_{11}$) supporting $choose(box2)$:

$$\mathcal{A}_5 = \left\langle \begin{array}{l} choose(box2) \prec diff(box2, box1), pref(box2, box1) \\ pref(box2, box1) \prec nearer\_than(box2, box1, store), \\ \quad \sim smaller(box2, box1) \end{array} \right\rangle \quad \mathcal{A}_6 = \left\langle \begin{array}{l} choose(box2) \prec diff(box2, box3), pref(box2, box3) \\ pref(box2, box3) \prec smaller(box2, box3) \end{array} \right\rangle$$

$$\mathcal{A}_7 = \left\langle \begin{array}{l} choose(box2) \prec diff(box2, box3), pref(box2, box3) \\ pref(box2, box3) \prec self(khep1) \\ \quad nearer\_than(box2, box3, khep1), \\ \quad smaller(box2, box3) \end{array} \right\rangle \quad \mathcal{A}_8 = \left\langle \begin{array}{l} choose(box2) \prec diff(box2, box4), pref(box2, box4) \\ pref(box2, box4) \prec smaller(box2, box4) \end{array} \right\rangle$$

$$\mathcal{A}_9 = \left\langle \begin{array}{l} choose(box2) \prec diff(box2, box4), pref(box2, box4) \\ pref(box2, box4) \prec nearer\_than(box2, box4, store), smaller(box2, box4) \end{array} \right\rangle$$

$$\mathcal{A}_{10} = \left\langle \begin{array}{l} choose(box2) \prec diff(box2, box4), pref(box2, box4) \\ pref(box2, box4) \prec self(khep1), nearer\_than(box2, box4, khep1), smaller(box2, box4) \end{array} \right\rangle$$

$$\mathcal{A}_{11} = \left\langle \begin{array}{l} choose(box2) \prec diff(box2, box4), pref(box2, box4) \\ pref(box2, box4) \prec self(khep1), nearer\_than(box2, box4, khep1), nearer\_than(box2, box4, store), smaller(box2, box4) \end{array} \right\rangle$$

Finally, from $\mathcal{P}_2$, the answers for $choose(box1)$, $choose(box3)$ and $choose(box4)$ are NO.

4 REASONING ABOUT ROBOTS

This section describes how a robot can decide when there are more robots working in the same environment. That is, the robot has to reason about other robots' choices. This topic must be considered in the reasoning processes of the robot because the presence of other robots in the environment require a coordinated behavior among them. Let us consider the following example to see how to implement this issue in a direct way in a DeLP-program.

Example 3 *Consider the situation shown in Figure 1(c). This situation extends the one presented in Figure 1(b) in that there is a second robot (khep2) in the environment. The knowledge of the robots, referring to this particular scenario, will be represented with the DeLP-programs* $\mathcal{P}_{3.1} = (\Pi_{3.1}, \Delta_3)$ *for khep1 and* $\mathcal{P}_{3.2} = (\Pi_{3.2}, \Delta_3)$ *for khep2. The perception of*

the current situation for khep1 ($\Pi_{f3.1}$) *is presented in Figure 4(b). The perception of khep2,* $\Pi_{f3.2} = \Pi_{f3.1} - \{self(khep1)\} \cup \{self(khep2)\}$. *The defeasible and strict rules of* $\mathcal{P}_{3.1}$ *and* $\mathcal{P}_{3.2}$ *coincide with the ones of* $\mathcal{P}_2$ *(Example 2), i.e.,* $\Pi_{r3.1}$=$\Pi_{r3.2}$=Π_{r2}, *and* Δ_3=Δ_2.

Up to this point, *khep*1 is not able of modelling *khep*2′*s* preferences because all the rules in $\mathcal{P}_{3.1}$ take into account only *khep*1′*s* preferences. However, if we include in $\mathcal{P}_{3.1}$ the rules presented in Figure 5, *khep*1 is now able of considering *khep*2′*s* preferences. Rule (95) allows *khep*1 to determine if there are more robots in the environment. If there are more robots in the environment (in particular there is only one in this situation), with the predicate $choose\_other(R, X)$, *khep*1 is able of determining which box X will be selected by the robot R. To put it briefly, $choose\_other(khep2, X)$ calls the DeLP interpreter from *khep*1 with $\mathcal{P}_{3.2}$ and receives in X the box that *khep*2 will choose. In this way, using rules (96)-(99), *khep*1 is able to select a different box to the one chosen by the other robots, considering in addition its own preferences. The rules shown in Figure 5 are strict because the fact that a box X was chosen by other robot is a non-defeasible reason for not choosing X.

$other(R) \leftarrow robot(R), self(Z), diff(R, Z)$ (95)
$select\_ot(X) \leftarrow other(R), choose\_other(R, X)$ (96)
$\sim select\_ot(X) \leftarrow select\_ot(Y), diff(X, Y)$ (97)
$choose(X) \leftarrow diff(X, Y), pref(X, Y), \sim select\_ot(X)$ (98)
$\sim choose(X) \leftarrow diff(X, Y), pref(X, Y), select\_ot(X)$ (99)

Figure 5: Rules to consider other robot preferences

As it can be noted, *khep*1′*s* choice changes with respect to the selection made by *khep*1 in Example 2. Now, *khep*1 has additional information about the presence of other robot in the environment and the new rules presented in Figure 5 allow it to choose a different box. In this case, *khep*1 will choose *box*1 on the grounds that it beliefs that *khep*2 will choose *box*2, because it is its nearest smaller box near to the store, too. In this way, the overall performance is enhanced avoiding a conflict in the robots' selection choices. If we do not include in $\mathcal{P}_{3.1}$ rules to model other robots' preferences, both *khep*1 and *khep*2 will choose *box*2 as the box to be transported, because it is their smaller box near to themselves and near to the store. It is not possible to foresee which robot will transport the box, if any, because both can grab the box at the same time leading to an unexpected situation.

Let us consider the above-mentioned in more detail. From the knowledge of *khep*1, represented by the DeLP-program $\mathcal{P}_{3.1} = (\Pi_{3.1}, \Delta_3)$, the answer for $choose(box2)$ is NO because there are seven non-defeated arguments $\mathcal{A}_{12}$-$\mathcal{A}_{18}$ supporting $\sim choose(box2)$. To derive $\sim choose(box2)$, rules (40), (95), (96) and (99) are used in $\mathcal{A}_{12}$, but rules (95), (96) and (99) do not appear in $\mathcal{A}_{12}$ because they are strict. The same occurs with $\mathcal{A}_{13}$-$\mathcal{A}_{18}$, all of them use rules (95), (96) and (99) but only the defeasible rules are shown in these arguments. In some arguments the defeasible rules used to derive the preference between two different boxes, are the same, *e.g.*, $\mathcal{A}_{13}$ and $\mathcal{A}_{15}$ use rule (36), while in other cases, different defeasible rules are used to determine the same preference between two boxes *e.g.*, $\mathcal{A}_{13}$ and $\mathcal{A}_{14}$.

$$\mathcal{A}_{12} = \{\ pref(box2, box1) \prec nearer\_than(box2, box1, store), \sim smaller(box2, box1)\ \}$$

$$\mathcal{A}_{13} = \{\ pref(box2, box3) \prec smaller(box2, box3)\ \}$$

$$\mathcal{A}_{14} = \{\ pref(box2, box3) \prec self(khep1), nearer\_than(box2, box3, khep1), smaller(box2, box3)\ \}$$

$$\mathcal{A}_{15} = \{\ pref(box2, box4) \prec smaller(box2, box4)\ \}$$

$$\mathcal{A}_{16} = \{\ pref(box2, box4) \prec nearer\_than(box2, box4, store), smaller(box2, box4)\ \}$$

$$\mathcal{A}_{17} = \{\ pref(box2, box4) \prec self(khep1), nearer\_than(box2, box4, khep1), smaller(box2, box4)\ \}$$

$$\mathcal{A}_{18} = \{\ pref(box2, box4) \prec self(khep1), nearer\_than(box2, box4, khep1), nearer\_than(box2, box4, store), smaller(box2, box4)\ \}$$

Besides, the answer for $choose(box1)$ is YES because there are two non-defeated arguments($\mathcal{A}_{26}$ and $\mathcal{A}_{27}$) supporting this query. It is important to note, that arguments $\mathcal{A}_{26}$ and $\mathcal{A}_{27}$ are obtained using rules (36), (96)-(98) but rule (36) is the only one appearing in the arguments because the other ones are strict. Although, arguments $\mathcal{A}_{21}$-$\mathcal{A}_{25}$ also support $choose(box1)$ they have two blocking defeaters ($\mathcal{A}_{19}$ and $\mathcal{A}_{20}$) supporting $\sim choose(box1)$. The argumentation lines for the query $choose(box1)$ are shown in Figure 6. (Black triangles represent defeated arguments, white triangles non-defeated ones and dotted arrows the blocking defeat relation.) Furthermore, the answers for $choose(box3)$ and $choose(box4)$ are NO, as expected.

$$\mathcal{A}_{19} = \left\{ \begin{array}{l} \sim choose(box1) \prec diff(box1, box2), \sim pref(box1, box2) \\ \sim pref(box1, box2) \prec \sim smaller(box1, box2) \end{array} \right\}$$

$$\mathcal{A}_{20} = \left\{ \begin{array}{l} \sim choose(box1) \prec diff(box1, box2), \sim pref(box1, box2) \\ \sim pref(box1, box2) \prec nearer\_than(box2, box1, store), \sim smaller(box1, box2) \end{array} \right\}$$

$$\mathcal{A}_{21} = \left\{ \begin{array}{l} choose(box1) \prec diff(box1, box3), pref(box1, box3) \\ pref(box1, box3) \prec smaller(box1, box3) \end{array} \right\}$$

$$\mathcal{A}_{22} = \left\{ \begin{array}{l} choose(box1) \prec diff(box1, box3), pref(box1, box3) \\ pref(box1, box3) \prec self(khep1), nearer\_than(box1, box3, khep1), smaller(box1, box3) \end{array} \right\}$$

$$\mathcal{A}_{23} = \left\{ \begin{array}{l} choose(box1) \prec diff(box1, box3), pref(box1, box3) \\ pref(box1, box3) \prec self(khep1), nearer\_than(box1, box3, khep1), nearer\_than(box3, box1, store), smaller(box1, box3) \end{array} \right\}$$

$$\mathcal{A}_{24} = \left\{ \begin{array}{l} choose(box1) \prec diff(box1, box4), pref(box1, box4) \\ pref(box1, box4) \prec smaller(box1, box4) \end{array} \right\}$$

$$\mathcal{A}_{25} = \left\{ \begin{array}{l} choose(box1) \prec diff(box1, box4), pref(box1, box4) \\ pref(box1, box4) \prec self(khep1), nearer\_than(box1, box4, khep1), smaller(box1, box4) \end{array} \right\}$$

$$\mathcal{A}_{26} = \{ \; pref(box1, box3) \prec smaller(box1, box3) \; \} \quad \mathcal{A}_{27} = \{ \; pref(box1, box4) \prec smaller(box1, box4) \; \}$$

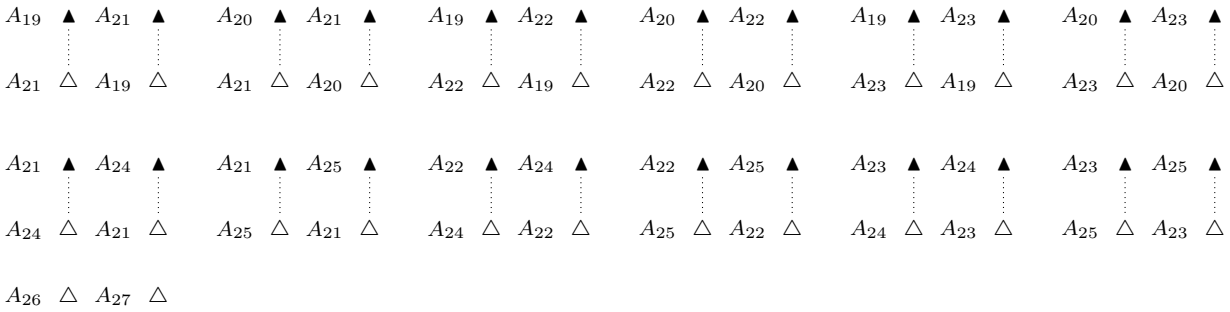

Figure 6: Argumentation lines for $choose(box1)$

Now, if we consider the decision from the $khep2's$ standpoint. From $\mathcal{P}_{3.2} = (\Pi_{3.2}, \Delta_3)$, using rule (46) a non-defeated argument supporting $choose(box2)$ can be built. Moreover, as expected, the answers for $choose(box1)$, $choose(box3)$ and $choose(box4)$ are NO.

It must be pointed out, that the robots coordination is effectively achieved assuming that $khep2$ does not simultaneously take into account $khep1's$ decisions (rules in Figure 5 are not included in $\mathcal{P}_{2.2}$). If this happen, both robots will avoid to choose $box2$ and they will take suboptimal decisions. In a context of recursive modeling this is equivalent to assume that $khep2$ is a 0-level agent that does not recognize the existence of other agents in the world.

5 RELATED WORK

After a proliferation period of a wide variety of reactive robotic architectures [5, 25, 19, 2], it was clear the necessity of introducing high-level deliberative processes in the decision making of autonomous robots. Regardless of the fact that deliberation has many advantages for decision making of an agent, it has the disadvantage of being slow compared to generating actions in a reactive fashion. Consequently, several hybrid architectures that combine the advantages of reactive and goal-directed aspects, were proposed having as their main difference the way they incorporate the deliberative component.

Arkin [1] was among the first to advocate the use of both deliberative and reactive control systems within the autonomous robot architecture, incorporating a traditional planner that could reason over flexible and modular reactive control system. Gat [12] proposed a three-level hybrid system [11] (Atlantis) incorporating a Lisp-based deliberator, a sequencer that handled failures of the reactive system, and a reactive controller. Estlin *et al.* [7] presented a two-layered architecture where the top decision layer contains techniques for autonomy creating a plan of robot commands, and the bottom functional layer provides standard robot capabilities that interface to system hardware. The main attention focus of the above-mentioned works has been the definition of the architecture related components necessary to achieve a successful behavior of the robots in real-life complex problems. Aspects like the description of the reactive component, the support of planning capabilities and the interaction of both components in an adequate planning-execution system for the robots, are considered in detail. Our work adopts a more top-down approach and concentrates on the high-level decision capabilities of the robots, an aspect that, in our opinion, is not sufficiently analyzed in the majority of these works. We consider that problems related to reactive control are important, but more attention should be paid to the deliberative processes involved in the robot's decision making, the main concern of the present paper.

Our proposal is closely related to the approach adopted by Parsons *et al.* [21]. This work incorporates a BDI deliberative component based on the work of Bratman on practical reasoning [4], where the internal state of an agent is determined by its knowledge about the environment (beliefs), the action facilities the agent is able to choose from (desires) and the current goals (intentions). In particular, in our work we follow some of the ideas exposed by Parsons *et al.* about the convenience of integrating 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 from the high-level component goals to be achieved, and plans to reach that goals are internally generated, and then an acknowledgement is received to inform failures or if everything finished as planned. Nonetheless, our work has some differences with the proposal of Parsons *et al.* 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. In [3], an agent called *Drama* incorporates an argumentation component which provides the ability to make flexible and context dependent decisions about medical treatment, based on several information sources (perspectives). The influence of different contexts that arise in changing environments is also considered in [15] where an argumentation-based framework supports the decision making of an agent modular architecture. In this case, arguments and their strength depend on the particular context that the agent finds himself. The fundamental role of argumentation for the management of uncertainty in symbolic decision making is

highlighted in [20] where several applications based on argumentative approaches are presented as empirical evidence of this claim. It is important to note that these argumentation systems have been usually integrated in *software* agents. Although in this paper we presented simulated scenarios, in our approach defeasible argumentation is applied in a robotic domain where the 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.

6 CONCLUSIONS AND FUTURE 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. The simple application domain described consists of different scenarios where simulated robots perform cleaning tasks. We have presented problems and their solutions, when there is only one robot in the environment, and when more than one robot are working in the same environment.

Future work includes considering more complex environments where robots have different sensing and acting capabilities. A deeper study and formalization of the process of implementing preference relations using an argumentative approach is required. Other important issue we are considering, is to extend this work to more general environments, where different levels of modeling of other robots are considered in the argumentation process, as well as their impact in the robots coordination.

REFERENCES

[1] R. C. Arkin. Integrating behavioral, perceptual and world knowledge in reactive navigation. *Robotics and Autonomous Systems*, 6:105–122, 1990.

[2] R. C. Arkin. *Behaviour-Based Robotics*. The MIT Press, 1998.

[3] Katie Atkinson, Trevor J. M. Bench-Capon, and Sanjay Modgil. Argumentation for decision support. In *DEXA*, pages 822–831, 2006.

[4] Michael E. Bratman. *Intention, Plans, and Practical Reason*. The David Hume Series. CSLI Publications, Stanford University, 1999.

[5] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, 1986.

[6] Jon Doyle and Richmond H. Thomason. Background to qualitative decision theory. *AI Magazine*, 20(2):55–68, 1999.

[7] T. Estlin, R. Volpe, I. Nesnas, D. Muts, F. Fisher, B. Engelhardt, and S. Chien. Decision-making in a robotic architecture for autonomy. In *International Symposium, on AI, Robotics and Automation for Space*, Montreal, Canada, June 2001.

[8] Edgardo Ferretti, Marcelo Errecalde, Alejandro García, and Guillermo Simari. Khedelp: A framework to support defeasible logic programming for the khepera robots. In *International Symposium on Robotics and Automation*, pages 98–103, 2006.

[9] Edgardo Ferretti, Marcelo Errecalde, Alejandro García, and Guillermo Simari. Khepera robots with argumentative reasoning. In *4th International Symposium on Autonomous Minirobots for Research and Edutainment*, 2-5, October 2007. ACCEPTED.

[10] Alejandro Javier García and Guillermo Ricardo Simari. Defeasible logic programming: an argumentative approach. *Theory and Practice of Logic Programming*, 4(2):95–138, 2004.

[11] E. Gat. Integrating planning and reacting in a heterogeneous asynchronous architecture for mobile robots. *SIGART Bulletin*, 2:70–74, 1991.

[12] E. Gat. On three-layer architectures. In *Artificial Intelligence and Mobile Robots*, 1998.

[13] Richard C. Jeffrey. *The Logic of Decision*. University Of Chicago Press, 2nd edition, 1990.

[14] K-Team. Khepera 2. http://www.k-team.com, 2002.

[15] Antonis Kakas and Pavlos Moraitis. Argumentation based decision making for autonomous agents. In *AAMAS*, pages 883–890, 2003.

[16] V. Lifschitz. Foundations of logic programming. In G. Brewka, editor, *Principles of Knowledge Representation*, pages 69–127. CSLI, 1996.

[17] Andreu Mas-Collel, Michael D. Whinston, and Jerry R. Green. *Microeconomic Theory*. Oxford University Press, 1995.

[18] Olivier Michel. Webots: Professional mobile robot simulation. *Journal of Advanced Robotics Systems*, 1(1):39–42, 2004.

[19] Stefano Nolfi and Dario Floreano. *Evolutionary Robotics*. The MIT Press, 2000.

[20] Simon Parsons and John Fox. Argumentation and decision making: A position paper. In *FAPR*, pages 705–709, London, UK, 1996. Springer-Verlag.

[21] Simon Parsons, Ola Pettersson, Alessandro Saffiotti, and Michael Wooldridge. *Artificial Intelligence Today: Recent Trends and Developments*, chapter Robots with the Best of Intentions, pages 329–338. Springer, 1999.

[22] A. Rao and M. Georgeff. Decision procedures for bdi logics. *Journal of Logic and Computation*, 8:293–342, 1998.

[23] Nicolás Rotstein, Alejandro García, and Guillermo Simari. Reasoning from desires to intentions: A dialectical framework. In *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07)*, July 2007. ACCEPTED.

[24] Guillermo R. Simari and Ronald P. Loui. A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence*, 53(2–3):125–157, 1992.

[25] V.Braitenberg. *Vehicles: Experiments in Synthetic Psychology*. The MIT Press, 1984.