# Multi-criteria Argumentation-Based Decision Making within a BDI Agent

Cecilia Sosa Toranzo, Marcelo Errecalde, and Edgardo Ferretti

Laboratorio de Investigación y Desarrollo en Inteligencia Computacional
Universidad Nacional de San Luis, Ejército de los Andes 950, San Luis - Argentina
e-mails:{csosatoranzo, merreca, ferretti}@unsl.edu.ar

**Abstract.** The BDI model, as a practical reasoning architecture aims at making decisions about what to do based on cognitives notions as beliefs, desires and intentions. However, during the decision making process, BDI agents also have to make background decisions like choosing what intention to achieve next from a set of possibly conflicting desires; which plan to execute from among the plans that satisfy a given intention; and whether is necessary or not to reconsider current intentions. With this aim, in this work, we present an abstract framework which integrates a *Possibilistic Defeasible Logic Programming* [1] approach to decision making in the inner decision processes within BDI agents.

**Keywords:** Agreement Technologies, Multi-criteria Decision Making, BDI, Argumentation, Possibilistic Defeasible Logic Programming

## 1 Introduction

The BDI model is a particular decision making model based on cognitive notions, namely: Belief, Desires and Intentions. This model is very relevant because of its similarity with human reasoning, the theoretical underpinning it has [2, 3], as well as its applicability to solve real-world problems [4, 5].

BDI architecture is inspired from Bratman's work on *practical reasoning* [2]. Practical reasoning (PR) aims at deciding what to do in a given situation and thus is directed towards action. However, besides deciding which action perform next, BDI agents also have to decide: (a) from a set of possibly conflicting desires which intention to achieve, (b) which plan execute from among the plans that satisfy the chosen intention, and (c) whether is necessary or not to reconsider current intentions. That is, BDI model also implies making background decisions.

Some of the issues mentioned above have been tackled in previous works. Casali *et al.* [6] present a general framework to define graded BDI agent architectures, where degrees in BDI models are used to set different levels of preferences or rejections on desires and preferences at intentions level to model the cost/benefit trade-off of reaching a goal. In [7], ideas from argumentation are combined with desire and planning rules, to give a formal account on how consistent sets of intentions can be obtained from a conflicting set of desires. A general framework for practical reasoning based on an abstract argumentative

machinery is presented in [8]. To the best of our knowledge, at present, there are no proposals which clearly formulate how these choices are made in BDI agent's inner decision processes. In this way, the main goal of this paper aims at incorporating in a generic way, multi-criteria decision making in BDI agent's inner decision processes. In particular, an argumentation-based approach to multi-criteria decision making is used [9]. In this respect, some proposals exist [10, 11], aiming at incorporating argumentation-based approaches within BDI agents.

The rest of the paper is organized as follows. Sect. 2 briefly introduces the BDI model to provide the background concepts underlying the proposed abstract framework (Sect. 3), which integrates multi-criteria argumentation-based decision making in the inner decision processes of the BDI architecture. Then, this framework is exemplified in the TILEWORLD domain (Sect. 4). Finally, Sect. 5 draws the conclusions and briefly describes possible future work.

## 2 BDI Model

Belief-Desires-Intentions models (BDI) have been inspired from the philosophical tradition on understanding *practical reasoning* and were originally proposed by Bratman *et al.* [2]. This kind of reasoning can be conceived as the process of deciding what action perform next to accomplish a certain goal. Practical reasoning involves two important processes, namely: deciding *what* states of the world to achieve and *how* to do it. The first process is known as *deliberation* and its result is a set of intentions. The second process, so-called *means-ends reasoning* involves generating actions sequences to achieve intentions.

The mental attitudes of a BDI agent on its beliefs, desires and intentions, represent its informational state, motivational state and decision state, respectively. The BDI architecture defines its cognitive notions as follows:

- **Beliefs:** Partial knowledge the agent has about the world.
- **Desires:** The states of the world that the agent would ideally like to achieve.
- **Intentions:** Desires (states of the world) that the agent has committed (dedicated resources) to achieve.

These cognitive notions are implemented as data structures in the BDI architecture, which also has an interpreter in charge of manipulating them to select the most appropriate actions to be performed by the agent. This interpreter performs the deliberation and means-ends reasoning processes aforementioned, and its simpler version is shown in Algorithm 1, as proposed in [12].

A usual problem in designing practical reasoning agents lies in getting a good balance among deliberation, means-ends reasoning and actions execution. It is clear that, in some point of time, an agent should drop some of its intentions, because they were already achieved, they are impossible to be achieved or makes no sense to do it, etc. Likewise, when opportunities arise to achieve new desires, the agent should generate intentions aiming at accomplishing them. Thus, as mentioned above it is important for an agent to *reconsider its intentions*. However, intentions reconsideration is costly in terms of time and computational resources.

**Algorithm 1** Agent control loop (version 1)

```
1: while true do
2:     observe the world;
3:     update internal world model;
4:     deliberate about what intention to achieve next;
5:     use means-ends reasoning to get a plan for the intention;
6:     execute the plan;
7: end while
```

Moreover, it can happen that some of the actions from the executing plan might fail in achieving the intended results, hence *replanning* capabilities should be provided. Both replanning and intentions reconsideration (if performed) must be carried out during the execution phase of the chosen actions.

## 3 Integration Framework

As mentioned above, the BDI model uses the cognitive notions of beliefs, desires and intentions to decide what to do, but also, inner decisions exist related to these high-level decisions which, in our view, have not been clearly detailed in previous works. That is why, in this section we propose an abstract framework which integrates multi-criteria argumentation-based decision making to solve inner decision making in a BDI agent.

In Sect. 2 it was referred that a BDI agent comprises two fundamental processes, namely, deliberation and means-ends reasoning, which are followed by a plan execution stage. Within these processes (deliberation, means-ends reasoning and execution) the following inner decisions can be made:

- Choice among conflicting desires: *deliberation* requires to commit to an intention from among conflicting desires.
- Choice between plans: during *means-ends reasoning* it might be necessary to choose from among plans which achieve the same intention, that is, deciding which action perform to achieve a particular intention.
- Intentions reconsideration: during the *execution* process (of only one plan or a mega-plan involving all the plans the agent has committed to) decisions should be made with respect to whether reconsider or not current intentions based on the dynamics of the environment, and if so, if new intentions should be adopted or current intentions should be dropped.

All in all, our BDI architecture will incorporate an *Inner Decision Making Component (IDMC)* which will make inner decisions with respect to the different alternatives and the multiple criteria provided to the agent. In our proposal, to select the best alternative from a given set of alternatives, the agent will have the $select(\cdot,\cdot,\cdot)$ function that will return the choice made by IDMC. This function will be used (within this framework) in all the inner decision processes a BDI agent has. It will receive as input parameters: (1) a set $B$ of candidate

alternatives, (2) the set $C$ containing the criteria that will be used to compare alternatives among each other, and (3) the preferences $\mathcal{P}$, composed by a preference order among criteria and a preference order among the possible values an alternative can take for each particular criterion. To select an alternative, this function implements the argumentation-based decision framework proposed in [9]. Therefore, next section briefly describes this framework and a pseudo-code of the $select(\cdot,\cdot,\cdot)$ function is presented.

### 3.1 The Argumentation-Based Decision Framework

The argumentation-based decision framework described in this section is formally related to the *choice-based approach* (CBA) to decision making, as stated in [9]. The CBA takes as primitive object the choice behaviour of the individual, which is represented by means of a *choice structure* ($\mathcal{B}$, $\mathbf{C}(\cdot)$) consisting of two elements:

- $\mathcal{B}$ is a set of subsets of $X$ (the set containing all the available alternatives to the decision maker). Each set $B \in \mathcal{B}$, represents a set of alternatives (or *choice experiment*) that can be conceivably posed to the decision maker.
- $\mathbf{C}(\cdot)$ is a *choice rule* 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 $B$. ($\mathbf{C}(B) \subseteq B$ for every $B \in \mathcal{B}$). When $\mathbf{C}(B)$ contains a single element, this element represents the *individual's choice* among the alternatives in $B$. The set $\mathbf{C}(B)$ might, however, contain more than one element and in this case they would represent the *acceptable alternatives* in $B$ for the decision maker.

This decision framework is conceptually composed by three components. The first component is set $X$. The second component, the epistemic component, represents the agent's knowledge and preferences, and the third one is the decision component. Formally, the argumentation-based decision framework is a triple $\langle X, \mathcal{K}, \Gamma \rangle$ where:

- $X$ is the set of all the *possible alternatives* that can be presented to the decision maker.
- $\mathcal{K}$ is the *epistemic component* of the decision maker (see Definition 4.5 from [9]). Formally, $\mathcal{K}$ is a 5-tuple, $\mathcal{K} = \langle \mathcal{C}, >_{\mathcal{C}}, ACC, \Pi, \Delta \rangle$ where:
  - $\mathcal{C}$ is a set of *comparison literals* representing the preference criteria that the decision maker will use to compare the elements in $X$. Let $C = \{C_1, \ldots, C_n\}$ $(n > 0)$ be the set of preference criteria that will be used to compare the elements in $X$, each criterion $C_i$ has associated a *comparision literal* $c_i(\cdot,\cdot)$ that states the preference between two alternatives of $X$, based on their attribute values. Then, $\mathcal{C}$= $\{c_1(\cdot,\cdot), \ldots, c_n(\cdot,\cdot)\}$.
  - $>_{\mathcal{C}}$ is a strict total order over the elements of $\mathcal{C}$. (Definition 4.2 from [9]).
  - $ACC$ is a user-specified *aggregation function* that aggregate necessity degrees. $ACC$ must satisfy specific properties (see [9]) and function $f^{+}_{\Phi}(\cdot)$ is defined from it. Here we will use:

$$ACC(\alpha_1, ..., \alpha_n) = [1 - \prod_{i=1}^{n}(1 - \alpha_i)] + k \max(\alpha_1, ..., \alpha_n) \prod_{i=1}^{n}(1 - \alpha_i) \quad (1)$$

with $k \in (0, 1)$, which has been shown in [9] to satisfy the desired properties to apply the framework.

* $\Pi$ is a set of *certain clauses*.
* $\Delta$ is a set of *uncertain clauses*.

$P(\Pi, \Delta)$ is a conformant P-DeLP program (see Definition 4.3 from [9]).

– $\Gamma$ is the *decision component*. It is a set with two decision rules:[1]

$$\Gamma = \left\{ \begin{array}{l} DR1 : \{W\} \stackrel{B}{\Leftarrow} \{bt(W,Y)\},\ not\{bt(Z,W)\} \\ DR2 : \{W,Y\} \stackrel{B}{\Leftarrow} \{sp(W,Y)\},\ not\{bt(Z,W)\} \end{array} \right\} \text{ with } B \subseteq X.$$

Rule DR1 states that an alternative $W \in B$ will be chosen, if $W$ is better than another alternative $Y$ and there is not a better alternative $Z$ than $W$. Besides, rule DR2 says that two alternatives $W$, $Y \in B$ with the same properties will be chosen if there is not a better alternative $Z$ than $W$ and $Y$.

Let $B \in \mathcal{B}$ be a set of alternatives posed to the agent and $\langle X, \mathcal{K}, \Gamma \rangle$ be the agent's decision framework. Let $\{D_i \stackrel{B}{\Leftarrow} P_i, not\ T_i\}_{i=1...n} \subseteq \Gamma$ be the set of applicable decision rules with respect to $\mathcal{K}$. The set of *acceptable alternatives* of the agent will be $\Omega_B = \bigcup_{i=1}^{n} D_i$.

In Algorithm 2, a general algorithm which implements a choice rule $\mathbf{C}(\cdot)$ is presented. As it can be observed function $\mu$ has as input parameter a choice experiment ($B$). A choice experiment is a set containing at least one element, hence, this function returns *failure* if receives as argument an empty set (step 1). If the choice experiment has one element, then it is thus returned as solution since there is only one trivial choice to be made (step 2). Then, if a non-empty set was received as parameter, the resulting set *sol* is initialized (step 3) and a local copy (*ch*) of the original choice experiment is made (step 4). The computing process to determine the set of acceptable alternatives ends when *ch* becomes empty (step 6), thus exiting the main loop (step 5) returning the computed set of acceptable alternatives *sol* (step 13). While there are alternatives in *ch*, an alternative is removed from this set and is assigned to $h$ (step 7). If there is not a better alternative than $h$ in the choice experiment (step 9) and $h$ is better than any other alternative in the choice experiment (step 8), then $h$ is added to the resulting set *sol* (step 10), otherwise is discarded (step 9). Besides, if $h$ is not better than any other alternative in the choice experiment (step 8), but there is no other alternative (let us denoted it as $h'$) in the choice experiment better than $h$ (step 11), then it holds that $h$ and $h'$ have the same properties, and they are the best, therefore $h$ is added to the resulting set *sol* (step 12). It is worth mentioning, that in turn (when selected in step 7) $h'$ will also be added to *sol*.

Based on the above-mentioned framework, function $select(\cdot, \cdot, \cdot)$ executes the steps shown in Algorithm 3 to choose from among the alternatives in $B$.

[1] Due to space constraints, the literals $better(\cdot, \cdot)$ and $same\_prop(\cdot, \cdot)$ in [9], will be referred in this paper as $bt(\cdot, \cdot)$ and $sp(\cdot, \cdot)$, respectively.

**Algorithm 2** Compute Acceptable Alternatives

**function** $\mu$(choice-experiment) **returns** non-empty-set-of-alternatives, or failure
1: **if** EMPTY?(choice-experiment) **then return** failure
2: **if** SINGLETON?(choice-experiment) **then return** choice-experiment
3: $sol \leftarrow \emptyset$
4: $ch \leftarrow$ choice-experiment
5: **loop do**
6: **if** EMPTY?($ch$) **then exit**
7: $h \leftarrow$ REMOVE-ELEMENT($ch$)
8: **if** IS-$h$-BETTER-THAN-ANY-OTHER?(choice-experiment) **then**
9: **if** ANY-BETTER-THAN-$h$?(choice-experiment) **then** discard $h$
10: **else** ADD-ELEMENT($sol$,$h$)
**else**
11: **if** ANY-BETTER-THAN-$h$?(choice-experiment) **then** discard $h$
12: **else** ADD-ELEMENT($sol$,$h$)
13: **return** $sol$

**Algorithm 3** Computation for alternatives selection

**function** *select*(alternatives $B$, criteria $C$, preferences $\mathcal{P}$) **returns** non-empty-set-of-alternatives, or failure

1: Define the comparison literal for each $C_i \in C$
2: Define $>_C$ according to the preferences of each criterion in $\mathcal{P}$
3: Build a conformant program $P(\Pi, \Delta)$ (as defined in [9])
4: **return** Evaluation of function $\mu(B)$

## 4 Example: The Tileworld

The TILEWORLD experimental domain [13] is a grid environment containing agents, tiles, holes and obstacles. The agent's objective consists of scoring as many points as possible by pushing the tiles into the holes to fill them in. The agent is able to move up, down, left, or right, one cell at a time, having as only restriction that obstacles must be avoided. This environment is dynamic, so that holes and tiles may randomly appear and disappear in accordance to a series of world parameters, which can be varied by the experimenter.

A BDI agent for the TILEWORLD can be implemented as follows: the agent's beliefs consist of its perceptions about the objects locations, as well as the score and time-out time for all the holes. Desires are the holes to be filled in, and the current intention (IH) aims at filling a particular hole right now. The means-end reasoner basically is a special-purpose route planner, which guides the agent to a particular tile that must be pushed into the hole to be filled in. Figure 1 shows a hypothetical scene in which the framework proposed in Sect. 3 will be used.

The agent gets its perception and updates its beliefs, in order to deliberate about what intention to achieve next. During deliberation it gets its reachable holes (options), *i.e.,* those which are not surrounded by obstacles and their time-out times are higher or equal to the distances from the agent to the holes. Then,

filtering stage takes place where one of the reachable holes is selected and becomes $IH$. In this case, all options are conflicting each other, since it is not possible to fill in more than one hole at a time. Hence, all reachable holes will serve as input to $selec(\cdot,\cdot,\cdot)$ function. In this way, $B = \{h_3, h_4, h_5\}$, $C = \{C_1 = \text{score}, C_2 = \text{timeout}, C_3 = \text{distAgent}, C_4 = \text{tileAvail (distance to the nearest tile)}\}$ and $>_C = \{(distAgent, timeout), (distAgent, tileAvail), (timeout, tileAvail), (score, distAgent), (score, timeout), (score, tileAvail)\}$.

Figure 2 presents preference for each criterion. Table 1 shows the alternatives and their respective values for each criterion. Likewise, following the approach from [9], a conformant P-DeLP program would be:

$$\Delta = \left\{ \begin{array}{ll} (score(h_4,h_3),0.92) & (bt(W,Y) \leftarrow score(W,Y),0.99) \\ (score(h_4,h_5),0.83) & (\sim bt(W,Y) \leftarrow score(Y,W),0.99) \\ (score(h_5,h_3),0.83) & (bt(W,Y) \leftarrow distAgent(W,Y),0.74) \\ (distAgent(h_3,h_5),0.62) & (\sim bt(W,Y) \leftarrow distAgent(Y,W),0.74) \\ (distAgent(h_3,h_4),0.67) & (bt(W,Y) \leftarrow timeout(W,Y),0.49) \\ (distAgent(h_5,h_4),0.54) & (\sim bt(W,Y) \leftarrow timeout(Y,W),0.49) \\ (timeout(h_5,h_3),0.37) & (bt(W,Y) \leftarrow tileAvail(W,Y),0.24) \\ (timeout(h_5,h_4),0.38) & (\sim bt(W,Y) \leftarrow tileAvail(Y,W),0.24) \\ (timeout(h_3,h_4),0.26) & \\ (tileAvail(h_3,h_5),0.08) & \\ (tileAvail(h_4,h_5),0.08) & \end{array} \right\}$$

$$\Pi = \left\{ (\sim bt(W,Y) \leftarrow sp(W,Y),1) \qquad (\sim bt(W,Y) \leftarrow sp(Y,W),1) \right\}$$

In the particular program presented above, the necessity degrees of the clauses belonging to $(\Pi, \Delta)$ were calculated as follows:

1. Normalize the alternatives' attribute values to interval $[0, 1]$ for all of the preference criteria (see Table 1).
2. Compare the alternatives among each other with respect to the normalized preference criteria (see first column of Table 2). The necessity degree of the clause is calculated as the absolute value of the remainder of their normalized attribute values.
3. Divide the necessity degrees obtained in previous step by the number of preference criteria provided to the decision maker, *i.e.,* by 4 in this case (see second column of Table 2).
4. Map the necessity degrees obtained in previous step to the subinterval assigned to the comparison literal in the clause (see third column of Table 2).
5. For each clause $(\varphi, \alpha)$ such that $\varphi$ is a rule of the kind $bt(W,Y) \leftarrow c_i(W,Y)$ or $\sim bt(W,Y) \leftarrow c_i(Y,W)$, set $\alpha$ to the upper bound value of the subinterval assigned to $c_i(\cdot,\cdot)$.

| Alternatives | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $C_{1[0,1]}$ | $C_{2[0,1]}$ | $C_{3[0,1]}$ | $C_{4[0,1]}$ |
|---|---|---|---|---|---|---|---|---|
| $h3$ | 3 | 8 | 2 | 2 | 0.33 | 0.53 | 0.33 | 0.67 |
| $h4$ | 9 | 7 | 6 | 2 | 1 | 0.47 | 1 | 0.67 |
| $h5$ | 6 | 15 | 5 | 3 | 0.67 | 1 | 0.83 | 1 |

**Table 1.** Alternatives values for each criterion

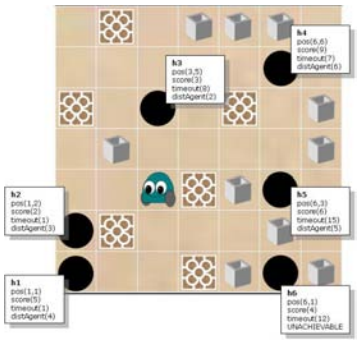

**Fig. 1.** Tileworld scene

| Criteria | Comparison literal | Subinterval |
|---|---|---|
| $C_1$ | $score(\cdot,\cdot)$ | $[0.75, 1)$ |
| $C_2$ | $timeout(\cdot,\cdot)$ | $[0.25, 0.5)$ |
| $C_3$ | $distAgent(\cdot,\cdot)$ | $[0.50, 0.75)$ |
| $C_4$ | $tileAvail(\cdot,\cdot)$ | $[0, 0.25)$ |

**Fig. 2.** Preferences per criterion

| | | |
|---|---|---|
| $(score(h_4,h_3),0.67)$ | $(score(h_4,h_3),0.17)$ | $(score(h_4,h_3),0.92)$ |
| $(score(h_4,h_5),0.33)$ | $(score(h_4,h_5),0.08)$ | $(score(h_4,h_5),0.83)$ |
| $(score(h_5,h_3),0.34)$ | $(score(h_5,h_3),0.08)$ | $(score(h_5,h_3),0.83)$ |
| $(distAgent(h_3,h_5),0.5)$ | $(distAgent(h_3,h_5),0.12)$ | $(distAgent(h_3,h_5),0.62)$ |
| $(distAgent(h_3,h_4),0.67)$ | $(distAgent(h_3,h_4),0.17)$ | $(distAgent(h_3,h_4),0.67)$ |
| $(distAgent(h_5,h_4),0.17)$ | $(distAgent(h_5,h_4),0.04)$ | $(distAgent(h_5,h_4),0.54)$ |
| $(timeout(h_5,h_3),0.47)$ | $(timeout(h_5,h_3),0.12)$ | $(timeout(h_5,h_3),0.37)$ |
| $(timeout(h_5,h_4),0.53)$ | $(timeout(h_5,h_4),0.13)$ | $(timeout(h_5,h_4),0.38)$ |
| $(timeout(h_3,h_4),0.06)$ | $(timeout(h_3,h_4),0.01)$ | $(timeout(h_3,h_4),0.26)$ |
| $(tileAvail(h_3,h_5),0.33)$ | $(tileAvail(h_3,h_5),0.08)$ | $(tileAvail(h_3,h_5),0.08)$ |
| $(tileAvail(h_4,h_5),0.33)$ | $(tileAvail(h_4,h_5),0.08)$ | $(tileAvail(h_4,h_5),0.08)$ |

**Table 2.** Alternatives comparison

The following arguments are built from the above P-DeLP conformant program:[2]

$\mathcal{A}_1 = \{(bt(h_4,h_3) \leftarrow score(h_4,h_3),0.99),\ (score(h_4,h_3),0.92)\}$
$\mathcal{A}_2 = \{(\sim bt(h_3,h_4) \leftarrow score(h_4,h_3),0.99),\ (score(h_4,h_3),0.92)\}$
$\mathcal{A}_3 = \{(bt(h_4,h_5) \leftarrow score(h_4,h_5),0.99),\ (score(h_4,h_5),0.83)\}$
$\mathcal{A}_4 = \{(\sim bt(h_5,h_4) \leftarrow score(h_4,h_5),0.99),\ (score(h_4,h_5),0.83)\}$
$\mathcal{A}_5 = \{(bt(h_5,h_3) \leftarrow score(h_5,h_3),0.99),\ (score(h_5,h_3),0.83)\}$
$\mathcal{A}_6 = \{(\sim bt(h_3,h_5) \leftarrow score(h_5,h_3),0.99),\ (score(h_5,h_3),0.83)\}$
$\mathcal{A}_7 = \{(bt(h_3,h_5) \leftarrow distAgent(h_3,h_5),0.74),\ (distAgent(h_3,h_5)),0.62)\}$
$\mathcal{A}_8 = \{(\sim bt(h_5,h_3) \leftarrow distAgent(h_3,h_5),0.74),\ (distAgent(h_3,h_5)),0.62)\}$
$\mathcal{A}_9 = \{(bt(h_3,h_4) \leftarrow distAgent(h_3,h_4),0.74),\ (distAgent(h_3,h_4),0.67)\}$
$\mathcal{A}_{10} = \{(\sim bt(h_4,h_3) \leftarrow distAgent(h_3,h_4),0.74),\ (distAgent(h_3,h_4),0.67)\}$
$\mathcal{A}_{11} = \{(bt(h_5,h_4) \leftarrow distAgent(h_5,h_4),0.74),\ (distAgent(h_5,h_4),0.54)\}$
$\mathcal{A}_{12} = \{(\sim bt(h_4,h_5) \leftarrow distAgent(h_5,h_4),0.74),\ (distAgent(h_5,h_4),0.54)\}$
$\mathcal{A}_{13} = \{(bt(h_5,h_3) \leftarrow timeout(h_5,h_3),0.49),\ (timeout(h_5,h_3),0.37)\}$
$\mathcal{A}_{14} = \{(\sim bt(h_3,h_5) \leftarrow timeout(h_5,h_3),0.49),\ (timeout(h_5,h_3),0.37)\}$
$\mathcal{A}_{15} = \{(bt(h_5,h_4) \leftarrow timeout(h_5,h_4),0.49),\ (timeout(h_5,h_4),0.38)\}$
$\mathcal{A}_{16} = \{(\sim bt(h_4,h_5) \leftarrow timeout(h_5,h_4),0.49),\ (timeout(h_5,h_4),0.38)\}$

[2] To simplify notation, given an argument $\langle \mathcal{A}, h, \alpha \rangle$, only the set $\mathcal{A}$ of uncertain clauses will be given since the conclusion $h$ and its necessity degree $\alpha$ can be obtained from it.

$\mathcal{A}_{17} = \{(bt(h_3, h_4) \leftarrow timeout(h_3, h_4), 0.49),\ (timeout(h_3, h_4), 0.26)\}$
$\mathcal{A}_{18} = \{(\sim bt(h_4, h_3) \leftarrow timeout(h_3, h_4), 0.49),\ (timeout(h_3, h_4), 0.26)\}$
$\mathcal{A}_{19} = \{(bt(h_3, h_5) \leftarrow tileAvail(h_3, h_5), 0.24),\ (tileAvail(h_3, h_5), 0.08)\}$
$\mathcal{A}_{20} = \{(\sim bt(h_5, h_3) \leftarrow tileAvail(h_3, h_5), 0.24),\ (tileAvail(h_3, h_5), 0.08)\}$
$\mathcal{A}_{21} = \{(bt(h_4, h_5) \leftarrow tileAvail(h_4, h_5), 0.24),\ (tileAvail(h_4, h_5), 0.08)\}$
$\mathcal{A}_{22} = \{(\sim bt(h_5, h_4) \leftarrow tileAvail(h_4, h_5), 0.24),\ (tileAvail(h_4, h_5), 0.08)\}$

To calculate the accrued structures for these arguments, it will be used the $ACC$ function defined below, with $K = 0.1$:[3]

$$ACC(\alpha_1, \ldots, \alpha_n) = [1 - \textstyle\prod_{i=1}^{n}(1 - \alpha_i)] + K \max(\alpha_1, \ldots, \alpha_n) \textstyle\prod_{i=1}^{n}(1 - \alpha_i)$$

As it can be observed, twelve a-structures can be built to support the reasons by which an alternative should be deemed better than another one.

$[\Phi_1, bt(h_3, h_5), 0.67]$, $[\mathbf{\Phi'_1}, \mathbf{\sim bt(h_3, h_5), 0.90}], \Phi_1 = \mathcal{A}_7 \cup \mathcal{A}_{19},\ \Phi'_1 = \mathcal{A}_6 \cup \mathcal{A}_{14};$
$[\Phi_2, \sim bt(h_5, h_3), 0.67], [\mathbf{\Phi'_2}, \mathbf{bt(h_5, h_3), 0.90}]$, $\Phi_2 = \mathcal{A}_8 \cup \mathcal{A}_{20},\ \Phi'_2 = \mathcal{A}_5 \cup \mathcal{A}_{13};$
$[\Phi_3, bt(h_3, h_4), 0.78]$, $[\mathbf{\Phi'_3}, \mathbf{\sim bt(h_3, h_4), 0.93}], \Phi_3 = \mathcal{A}_9 \cup \mathcal{A}_{17},\ \Phi'_3 = \mathcal{A}_2;$
$[\Phi_4, \sim bt(h_4, h_3), 0.78], [\mathbf{\Phi'_4}, \mathbf{bt(h_4, h_3), 0.93}]$, $\Phi_4 = \mathcal{A}_{10} \cup \mathcal{A}_{18}, \Phi'_4 = \mathcal{A}_1;$
$[\Phi_5, bt(h_5, h_4), 0.73]$, $[\mathbf{\Phi'_5}, \mathbf{\sim bt(h_5, h_4), 0.85}], \Phi_5 = \mathcal{A}_{11} \cup \mathcal{A}_{15}, \Phi'_5 = \mathcal{A}_4 \cup \mathcal{A}_{22};$
$[\Phi_6, \sim bt(h_4, h_5), 0.73], [\mathbf{\Phi'_6}, \mathbf{bt(h_4, h_5), 0.85}]$, $\Phi_6 = \mathcal{A}_{12} \cup \mathcal{A}_{16}, \Phi'_6 = \mathcal{A}_3 \cup \mathcal{A}_{21};$

Those a-structures warranted from the dialectical process (shown in bold), will be used by Algorithm 2 to compute the set of acceptable alternatives. In this particular case, only decision rule DR1 can be applied. For alternative $h_4$, precondition of DR1 can be warranted and like there is no warranted a-structure supporting a conclusion of the kind $bt(Z, h_4)$ to warrant DR1's restriction, $h_4$ becomes the acceptable alternative. Finally, hole $h_4$ becomes $IH$.

Once a hole has been selected to fill in, plans to achieve this intention are selected. The criteria set provided for plan selection could be $C = \{C_1 = length$, $C_2 = \text{cost}$, $C_3 = \text{timeoutTile}\}$. Criterion $C_1$ is the number of action within the plan. $C_2$ represents the plan cost which is calculated as the sum of its actions costs, which depend on the agent's orientation. Finally, $C_3$ is the time-out time of the tile selected in the plan to fill in the hole.

On the other hand, the fact that holes appear and disappear causes the agent to change its intentions. For example, when the set of holes dot not change while the agent is executing a plan, then there is no need to deliberate; but if the set of holes do change, this might mean that $IH$ has disappeared or that a closer hole has appeared; thus, intentions reconsideration is necessary. To achieve this behaviour, it is important to consider appropriate criteria to determine whether these changes have occurred or not. Means-ends reasoning and intention reconsideration also use the argumentation-based decision framework (as in the filtering stage), in order to choose a plan to execute or to reconsider intentions, while the plan is under execution. Due to space constraints, how this framework is applied in these stages, will not be developed in this paper.

[3] This function is a variant of the One-Complement accrual function used in [14] where $K$ aims at weighting the importance given to the highest priority preference criterion.

## 5 Conclusions

In this work, we presented an abstract framework that integrates argumentation-based decision making from a multi-criteria approach, within the inner decision processes of a BDI agent. In this way, the contribution of this work is twofold. On one hand, it was specified how to perform concrete implementations of inner decision making processes within a BDI agent. On the other hand, different criteria and preferences were aggregated to get a solution to a multi-criteria decision problem as an instantiation of argumentation-based decision making.

In order to get a better understanding and provide feedback to the abstraction process carried out to propose this present framework, as future work, following the idea proposed in [15], new instantiations of the framework will be done with other methods belonging to the research field of Agreement Technologies.

## References

1. Alsinet, T., Chesñevar, C.I., Godo, L., Simari, G.: A logic programming framework for possibilistic argumentation: formalization and logical properties. Fuzzy Sets and Systems **159**(10) (2008) 1208–1228
2. Bratman, M., Israel, D., Pollack, M.: Plans and resource bounded reasoning. Computational Intelligence **4**(4) (1988) 349–355
3. Dennett, D.C.: Intentional systems. Journal of Philosophy **68** (1971) 87–106
4. Evertsz, R., Fletcher, M., Jones, R., Jarvis, J., Brusey, J., Dance, S.: Implementing Industrial Multi-agent Systems Using JACK. In: Programming Multi-Agent Systems. Springer (2004)
5. Benfield, S.S., Hendrickson, J., Galanti, D.: Making a strong business case for multiagent technology. In: 5th AAMAS. (2006)
6. Casali, A., Godo, L., Sierra, C.: A graded BDI agent model to represent and reason about preferences. Artifical Intelligence **175**(7-8) (2011) 1468–1478
7. Amgoud, L.: A formal framework for handling conflicting desires. In Nielsen, T.D., Zhang, N.L., eds.: ECSQARU. Volume 2711 of Lecture Notes in Computer Science., Springer (2003) 552–563
8. Amgoud, L., Prade, H.: Formalizing practical reasoning under uncertainty: An argumentation-based approach. In: IAT, IEEE Computer Society (2007) 189–195
9. Ferretti, E., Errecalde, M., García, A., Simari, G.: A possibilistic defeasible logic programming approach to argumentation-based decision making. Manuscript ID: TETA-2012-0093.R1. Under Review process in *JETAI*. `https://sites.google.com/site/edgardoferretti/TETA-2012-0093.R1.pdf?attredirects=0&d=1`.
10. Rotstein, N.D., García, A.J., Simari, G.R.: Reasoning from desires to intentions: A dialectical framework. In: AAAI, AAAI Press (2007) 136–141
11. Schlesinger, F., Ferretti, E., Errecalde, M., Aguirre, G.: An argumentation-based BDI personal assistant. In: IEA/AIE. Volume 6069 of LNAI., Springer (2010)
12. Wooldridge, M.: Reasoning about Rational Agents. The MIT Press (2000)
13. Pollack, M.E., Ringuette, M.: Introducing the tileworld: Experimentally evaluating agent architectures. In: 8th AAAI. (1990) 183–189
14. Gómez, M., Chesñevar, C., Simari, G.: Modelling argument accrual in possibilistic defeasible logic programming. In: ECSQARU. LNCS, Springer (2009) 131–143
15. Sosa-Toranzo, C., Schlesinger, F., Ferretti, E., Errecalde, M.: Integrating a voting protocol within an argumentation-based BDI system. In: XVI CACIC. (2010)