Intention Reconsideration like Dichotomous Choice: an Argumentation-based Approach

Cecilia Sosa Toranzo
LIDIC, Departamento de Informática
Universidad Nacional de San Luis
Email: csosatoranzo@unsl.edu.ar

Edgardo Ferretti
LIDIC, Departamento de Informática
Universidad Nacional de San Luis
Email: ferretti@unsl.edu.ar

Marcelo Errecalde
LIDIC, Departamento de Informática
Universidad Nacional de San Luis
Email: merreca@unsl.edu.ar

***Abstract*—A key issue in the design of Belief-Desire-Intention (BDI) agents is that of finding an appropriate Intention Reconsideration (IR) strategy. IR presents a dichotomous choice to decide between deliberating or acting. This choice typically involves multiple criteria, and thus IR can be seen as a multi-criteria decision-making problem. Traditional approaches to IR define the strategy in the agent's design stage, which makes it impossible to modify it in execution time. This is clearly not a practical solution for agents operating in changing environments. That is why, in this work, we propose implementing the IR strategy with an argumentation-based mechanism that uses arguments accrual in Possibilistic Defeasible Logic Programming. This approach allows to change commitments to intentions depending on how the environment evolves, and takes into consideration multiple criteria when making a decision whether to reconsider or not.**

***Index Terms*—BDI agents, dichotomous choice, intention reconsideration, possibilistic defeasible logic programming.**

I. Introduction

The BDI model, as a practical reasoning architecture aims at making decisions about what to do based on cognitive notions as Beliefs, Desires and Intentions. A very important design issue in BDI agents concerns defining the intention reconsideration strategy [1], [2]. This strategy defines under which circumstances the BDI agent will use computational resources to deliberate about its intentions. At present there is no consensus on when or how an agent should reconsider its intentions. Current proposals consider the agents' *commitment levels*, which range from *cautious* agents (which reconsider their intentions after each action execution) to *bold* agents (where no reconsideration is performed until the current plan has been completely executed).

In [3], the efficiency of these strategies in different kinds of environments is investigated, but the intention reconsideration strategy is defined in the agent's design stage, which makes impossible to modify this strategy in execution time. It is clear that this is not a practical solution for agents operating in dynamic and changing environments. In this respect, in [2], a framework is proposed that allows the agent choosing by itself what strategy to follow based on the current state of the world. The main idea underlying this work is that an intention reconsideration (IR) strategy can be conceived as a meta-level control process which selects whether to deliberate or to act. Similarly, in [4] a theoretical framework for the intention reconsideration problem based on Markov decision processes (MDPs) is presented, which involves the construction of a meta-level MDP in which the two actions are "think" or "act".

978-1-5386-3057-0/17/$31.00

In [2], the proposed model incorporates decision making within the IR process of a BDI agent. To determine the best possible action, the maximum expected utility is considered; that is, only one criterion is taken into account to solve the decision problem. However, IR typically involves considering multiple criteria. In this respect, Sosa-Toranzo *et al.* [5] extend [2] by applying voting-based multi-criteria decision making to choose whether to act or to deliberate. Besides, multi-criteria decision making is also applied to make other background decisions within the BDI architecture, like choosing among conflicting desires and choosing between plans.

In particular, in [5] a voting scheme is used to perform the above-mentioned inner decisions within the BDI architecture, but the proposed framework offers easy and coherent integration of different agreement technologies (ATs). In fact, as future work, Sosa-Toranzo *et al.* argue that they will focus on the "inter mechanism" comparisons of integrating different ATs at the different decision stages of the BDI architecture. This current work, represents a step forward to this aim since the dichotomous choice model proposed in [6] that is based on a voting scheme, is adapted to an argumentative approach that is used to instantiate the intentions reconsideration component. The key feature of this novel approach to IR, is that it allows to change commitments to intentions depending on how the environment evolves (in contrast to fixed strategies [3]), and given the nature of the dichotomous choice model, multiple decision criteria are considered. Besides, to get a better understanding of our proposal, a full example is depicted by using the TileWorld experimental domain [7]; also used in [5].

The rest of the paper is organized as follows. Section II briefly introduces the Belief-Desire-Intention models, to provide the background concepts underlying intention reconsideration, that is the primary issue of our work. Section III describes the TileWorld experimental domain that will be used as a running example throughout the paper. Then, Section IV proposes to interpret IR as an argumentative process that implements dichotomous choices. With that aim, this section also introduces a theoretical review of the argumentation formalism that will be used. Finally, Section V draws the conclusions and briefly describes possible future work.

II. BDI Model

Belief-Desire-Intention (BDI) models have been inspired from the philosophical tradition on understanding *practical reasoning* and were originally proposed by Bratman *et al.* [8]. 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 action sequences to achieve intentions, and is usually followed by a plan execution stage.

The mental attitudes of a BDI agent are its beliefs, desires and intentions, which 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 been committed (dedicated resources) to achieve.

These cognitive notions are implemented as data structures in the BDI architecture, which also has an interpreter that manipulates 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 it is shown in Algorithm 1 (as presented in [1]). It assumes that an explicit representation of desires (D), beliefs (B) and intentions (I) exists, within the agent. The agent's perceptions will be represented by ρ and regarding plans, they will be referred to as "recipes" to achieve intentions. Therefore, π will be used to denote plans. Lines $1-3$ initialize beliefs, intentions and the plan. The main control loop comprises lines $4-19$. In lines $5-6$, the agent perceives and updates its beliefs while in line 7 it decides whether to reconsider or not. In lines $8-9$ it deliberates and in line 11, it generates a plan to achieve its intentions. Lines $14-17$ show that an action of the current plan is executed. Because the purpose of the functions in this loop can be easily derived from their names, we omit their formalizations but the interested reader should refer to [1].

Algorithm 1 BDI Agent control loop

```
1:  B ← B_0
2:  I ← I_0
3:  π ← null
4:  while true do
5:     get next percept ρ
6:     update B on the basis of ρ
7:     if reconsider(B,I) then
8:        D ← options(B,I)
9:        I ← filter(B,D,I)
10:       if not sound(π,I,B) then
11:          π ← plan(B,I)
12:       end if
13:    end if
14:    if π ≠ ∅ then
15:       α ← hd(π)
16:       execute(α)
17:       π ← tail(π)
18:    end if
19: end while
```

A. Intention Reconsideration

Intentions can be adopted, maintained and modified. Intentions are maintained by means of a commitment strategy. Intention maintenance concerns the commitment to single intentions that have already been adopted. Intention modification happens "in the light of changing circumstances" through re-deliberation. Intentions resist change, but there are conditions under which a rational agent must consider whether its intentions are still worth committing to. That is, an agent should *reconsider its intentions* when it is reasonable to do so. However, IR is a computationally costly process, and is a kind of meta-level reasoning. It is therefore necessary to fix upon an IR strategy that makes optimal use of the available computational resources. Whereas a commitment strategy says when an individual intention should be kept or dropped, a reconsideration strategy says when to deliberate. Reconsideration is equivalent to re-deliberation. That is, when an agent decides to reconsider, it activates its deliberation process.[1]

To try capture the trade-off between deliberating or acting (continue with current intentions), Algorithm 1 incorporates an explicit meta-level control component: $reconsider(\cdot,\cdot)$ function. Possible interactions between deliberation and meta-level control (whether to deliberate again)[2] are summarized in Table I.

TABLE I: Interactions between $reconsider(\cdot,\cdot)$ and deliberation [8].

Situation number	Chose to deliberate?	Changed intentions?	Would've changed intentions?	$reconsider(\cdot,\cdot)$ optimal?
1	No		No	Yes
2	No		Yes	No
3	Yes	No		No
4	Yes	Yes		Yes

The analysis elucidates when reconsideration is optimal. In situation 1, the agent does not deliberate, but if it did, it would not have changed its intentions anyway. This situation is desirable. In situation 2, the agent does not deliberate, but if it did, it would have changed its intentions. Here the agent gets bad advice from $reconsider(\cdot,\cdot)$. The agent chooses to deliberate in situations 3 and 4. When it does not change its intentions in situation 3, the agent is wasting time deliberating.

[1] In Algorithm 1, deliberation process is considered as composed by $options(\cdot,\cdot)$ and $filter(\cdot,\cdot,\cdot)$ functions.

[2] It is assumed that the cost of executing the reconsider function is much less than the cost of the deliberation process itself.

The $reconsider(\cdot,\cdot)$ function is not behaving optimally. The agent does change intentions in situation 4, which means it was a good idea to deliberate, and $reconsider(\cdot,\cdot)$ has done well. In conclusion, the purpose of $reconsider(\cdot,\cdot)$ function is to deliberate when it pays off to deliberate (*i.e.,* when deliberation will lead to changing intentions), and otherwise not to deliberate, but to act.

The acquainted reader must have noticed that the stages mentioned above, viz. deliberation, means-ends reasoning and execution, involve making decisions, as described below:

- **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 actions 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.

III. The TileWorld domain

The TileWorld 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. Figure 1 shows a TileWorld's hypothetical scene in which the agent must select a hole to fill in, go to a tile, push it and cover the hole.

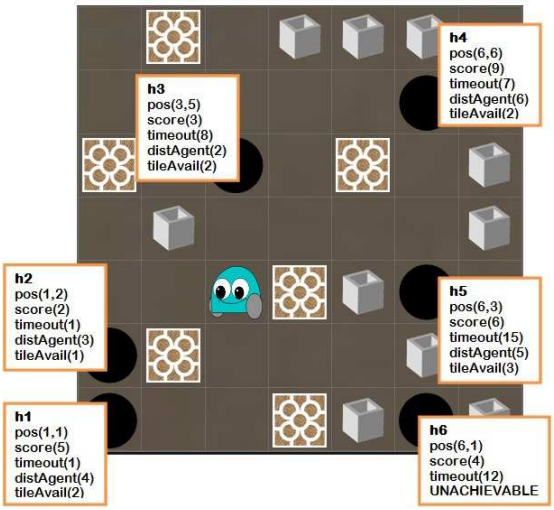

Fig. 1: TileWorld scene.

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 represent the holes to be filled in, and the current intention (IH) is filling a particular hole right now. The means-end reasoner is basically a special-purpose route planner, which guides the agent to a particular tile that must be pushed into the hole to be filled in. The agent's cycle is as follows: the agent gets its perception and then updates its beliefs. If $reconsider(\cdot,\cdot)$ function returns true, the agent should 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, the filtering stage takes place where one of the reachable holes is selected and becomes the IH. During the planning stage, a plan is selected to bring a tile to IH and the plan's first action (up, down, left, right) is then executed. Subsequently, the cycle begins again by obtaining a new perception of the world and evaluating if it is convenient to reconsider current intention (IH) or continue with the execution of the actions of the current plan. In this way, we can identify the three decision-making processes described at the end of Section II.

Choice among conflicting desires: The agent must select from the six holes which are present in the environment; $Holes = \{h_1, h_2, h_3, h_4, h_5, h_6\}$. These holes have several characteristics to take into account in order to make the decision: hole score (*score*), distance between the hole and the agent (*distAgent*), remaining lifetime of the hole (*timeout*), and how far is the closest tile to the hole (*tileAvail*). The dynamism of the environment makes impossible to the agent to decide solely based on the score of a hole, since the hole with the highest score (h_4) can disappear before the agent reaches it given its distance ($distAgent = 6$). A closer hole, h_3 ($distAgent = 2$) could be more reasonable to be filled in given that there is less chance of disappearance before the agent arrives and could also bring the agent to another hole with higher score (the agent would stay at a distance 4 from hole h_4). A hole as h_2, with a very close tile, is also a good option given that the agent would only need two moves to fill it in. Since there is no single criterion to decide which hole to fill next, the agent faces a multi-criteria decision making problem.

Choice between plans: Once the hole has been chosen, the agent should go to a tile and push it towards the selected hole. Since there are several tiles present in the environment (which may also appear and disappear) the agent can plan different ways to reach the hole based on the choice of different tiles. For example, as shown in Figure 2, the agent has planned four alternative ways to fill in hole h_5; $Plans = \{p_1, p_2, p_3, p_4\}$. Plan p_1 uses the tile located in position $pos(5,5)$ to cover the hole while plan p_2 uses the tile located in the position $pos(4,3)$. The choice of the plan to be implemented, as well

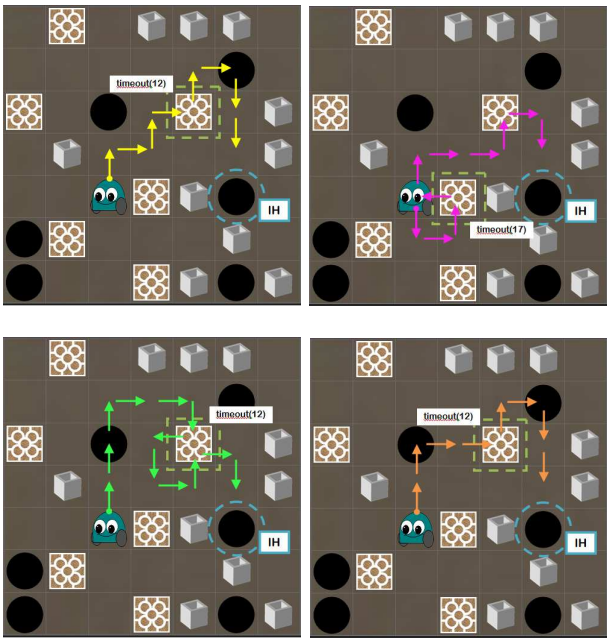

Fig. 2: Plans p_1, p_2, p_3, p_4.

as the choice of holes, present several criteria based on which to make the decision; such as: the number of cells that the agent must move to reach the hole ($length$), the cost of the plan ($cost$) and the remaining life time of the intervening tile in the plan ($tileAvail$).

Intentions reconsideration: Moreover, the appearance of a new hole could make it necessary to reconsider the current intention (to reach the previously selected hole) because if a hole appears, and is "better" than the hole that is currently being pursued, the agent should change its intention, *i.e.,* fill in the hole that appeared recently. In this case, the agent must decide whether to stop reconsidering his current intentions or continuing the execution of the actions of the plan that will lead it to the achievement of its current intention. Since the agent does not know the changes that will be presented in the environment, this choice is made under uncertainty considering the characteristics of the IH ($scoreIH, timeoutIH, distAgentIH$) and the characteristics known or learnt about the environment ($holeGestation$, $determinist$, $accessibility$). This situation presents a dichotomous choice between two alternatives (reconsider intentions or not) which is also a multi-criteria decision making problem, and next section shows how this problem can be solved from an argumentative perspective.

IV. Intention Reconsideration as an Argumentative Process

In [9], Wooldridge and Parsons proposed a formal model in which a BDI agent is explicitly equipped with a meta-level control strategy to select whether to deliberate or to act; that is to say, that the intention reconsideration strategy can be conceived as a meta-level control process. The underlying idea is that at any given instant, an agent has two available choices. It can either *deliberate* (that is, it can expend computational resources deciding wheter to change its focus), or it can act (that is, it can expend resources attempting to actually achieve its current intentions).[3] In this context, as argued in [6], intentions reconsideration presents a dichotomous choice either to choose between deliberating or acting, or between reconsidering versus not reconsidering. Therefore, the $reconsider(\cdot,\cdot)$ function called in line 7 of Algorithm 1, is implemented as shown in Algorithm 2. Since this implementation, is a modification to the algorithm proposed in [5], [10], a key issue that must be noted is the invocation to the $select(\cdot,\cdot,\cdot)$ function in line 8. In [5], this function returns the choice made by the Inner Decision Making Component (IDMC), the module incorporated in the proposed BDI architecture that is in charge of making all the inner decisions; viz. choosing among conflicting desires, choosing between plans and choosing whether to reconsider or not.

Algorithm 2 Intention Reconsideration

function: *reconsider*(beliefs B, intentions I) **return** boolean

1: C ← selection criteria
2: P ← user's preferences
3: get current plan π from I
4: **if** $\pi = \emptyset$ **then**
5: **return** true
6: **end if**
7: $A \leftarrow \{yes, no\}$
8: $a_{select} \leftarrow$ select(A, C, P)
9: **if** $a_{select} = yes$ **then**
10: **return** true
11: **end if**
12: **return** false

Hence, if we consider again the TileWorld example described in Section III, function $select$ will receive as first argument, alternatives $A = \{yes, no\}$. Then, as second argument we have the criteria set detailed below:

$$C = \left\{ \begin{array}{ll} C_1 = holeGestation, & C_2 = deterministic, \\ C_3 = accesibility, & C_4 = scoreIH, \\ C_5 = timeoutIH, & C_6 = distAgentIH \end{array} \right\}$$

where C_1 refers to how much the environment changes; C_2 refers to how deterministic the agent's actions are; C_3 specifies the agent's visibility percentage with respect to the entire environment; C_4 is the score of the IH, C_5 designates the remaining life-time of the IH; and C_6 is the distance from the agent to the IH. It is worth noticing that the subset $\{C_1, C_2, C_3\}$ refers to environmental criteria, and $\{C_4, C_5, C_6\}$ are criteria concerning the characteristics of the IH. Finally, the third argument would be the order of preference among the criteria considered, that in this example is: $C_1 \succ C_2 \succ C_5 \succ C_4 \succ C_3 \succ C_6$. It is clear that the

[3]It is worth noting that we assume the only way an agent can change its focus (i.e. modify its intentions) is through explicit deliberation.

smaller the values of criteria C_1, C_2, C_3, C_4, C_5 are, the more appropriate is to reconsider intentions. On the contrary, the higher the value of criterion C_6 is, the more appropriate is to reconsider intentions.

In Section IV-B we cover in detail, how the $select(\cdot,\cdot,\cdot)$ function is implemented by using argumentation, but in Section IV-A, some minimal theoretical background is introduced first.

A. P-DeLP overview

Possibilistic Defeasible Logic Programming (P-DeLP) is a language which combines features from argumentation theory, logic programming and a unified treatment of possibilistic uncertainty and fuzziness. Originally proposed in [11] it has been object of active research concerning its formalization. This subsection introduces the minimum necessary concepts of P-DeLP which are used in Section IV-B to pose intention reconsideration as a dichotomous choice based on argumentation. For a more comprehensive view on any topic related to this formalism, please refer to [12], [13].

1) P-DeLP Language and Arguments Accrual: In P-DeLP language, a literal "L" is a ground atom "A" or a negated ground atom "$\sim A$", where "$\sim$" represents the *strong negation* and atom comes from the terminology of Logic Programming [14]. Hence, literals have no variables. Strong negation [15] allows for the representation of conflictive or contradictory information.

A *weighted* clause is a pair (φ, α), where φ is a rule $q \leftarrow p_1 \wedge \ldots \wedge p_k (k \geq 0)$ or a fact q (*i.e.,* a rule with empty antecedent), where $q, p_1, \ldots, p_k$ are literals, and $\alpha \in [0,1]$ expresses a lower bound for the necessity degree of φ.[4] A clause (φ, α) is referred to as certain if $\alpha = 1$ and uncertain, otherwise. A set of P-DeLP clauses Γ is deemed *contradictory*, denoted $\Gamma \vdash \bot$, if, for some atom a, $\Gamma \vdash (a, \alpha)$ and $\Gamma \vdash (\sim a, \beta)$, with $\alpha > 0$ and $\beta > 0$, where $\vdash$ stands for deduction by means of the following instance of the *generalized modus ponens rule*:

$$\frac{(q \leftarrow p_1 \wedge \cdots \wedge p_k,\ \alpha) \quad (p_1, \beta_1), \ldots, (p_k, \beta_k)}{(q,\ \min(\alpha, \beta_1, \ldots, \beta_k))}\ [GMP]$$

Moreover, a P-DeLP program (see Definition 1) is a set of P-DeLP clauses in which certain information is distinguished from uncertain information, with the additional requirement that certain knowledge is required to be *non-contradictory*.

Definition 1: A P-DeLP program $\mathcal{P}$ (or just program $\mathcal{P}$) is a pair (Π, Δ), where Π is a non-contradictory finite set of certain clauses, and Δ is a finite set of uncertain clauses.

Since literals are ground, certain and uncertain clauses are also ground. However, following the usual convention [14], some examples will use schematic clauses with variables. Given a schematic clause R, $Ground(R)$ stands for the set of all ground instances of R. In order to distinguish variables from other elements of a schematic rule, we will denote variables with an initial upper-case letter.

With explanatory purposes, Figure 3 introduces the P-DeLP program whose design will be described in Section IV-B.

$$\mathcal{P} = \left\{ \begin{array}{l} (low(holeGestation), 0.92) \\ (low(distAgentIH), 0.03) \\ (high(deterministic), 0.82) \\ (high(accesibility), 0.32) \\ (high(scoreIH), 0.39) \\ (high(timeoutIH), 0.58) \\ (reconsider \leftarrow low(holeGestation), 0.99) \\ (reconsider \leftarrow low(deterministic), 0.82) \\ (reconsider \leftarrow low(accesibility), 0.32) \\ (reconsider \leftarrow low(scoreIH), 0.49) \\ (reconsider \leftarrow low(timeoutIH), 0.66) \\ (reconsider \leftarrow high(distAgentIH), 0.16) \\ (\sim reconsider \leftarrow high(holeGestation), 0.99) \\ (\sim reconsider \leftarrow high(deterministic), 0.82) \\ (\sim reconsider \leftarrow high(accesibility), 0.32) \\ (\sim reconsider \leftarrow high(scoreIH), 0.49) \\ (\sim reconsider \leftarrow high(timeoutIH), 0.66) \\ (\sim reconsider \leftarrow low(distAgentIH), 0.16) \end{array} \right\}$$

Fig. 3: P-DeLP program for the TILEWORLD domain described in Section III.

As it can be observed from Figure 3, the program contains no certain clauses since all the α values of the weighted clauses are lower than one. Besides, the first six clauses of the program correspond to facts given that they have empty antecedents. The remaining clauses are rules of the kind $q \leftarrow p_1$, that is $k = 1$ for all of them. Moreover, this program is contradictory because $\mathcal{P} \vdash (reconsider, 0.92)$ and $\mathcal{P} \vdash (\sim reconsider, 0.39)$, as shown below:

$$\frac{(reconsider \leftarrow low(holeGestation), 0.99) \quad (low(holeGestation), 0.92)}{(reconsider,\ \min(0.99, 0.92))}$$

$$\frac{(\sim reconsider \leftarrow high(scoreIH), 0.49) \quad (high(scoreIH), 0.39)}{(\sim reconsider,\ \min(0.49, 0.39))}$$

The notion of *argument* in P-DeLP (see Definition 2), refers to a tentative proof (as it relies to some extent on uncertain, possibilistic information) from a consistent set of clauses supporting a given conclusion h with a necessity degree α.

Definition 2: Given a P-DeLP program $\mathcal{P} = (\Pi, \Delta)$, a set $\mathcal{A} \subseteq \Delta$ of uncertain clauses is an *argument* for a conclusion h with necessity degree $\alpha > 0$, denoted $\langle \mathcal{A}, h, \alpha \rangle$, iff:

1) $\Pi \cup \mathcal{A}$ is non contradictory;

[4] The necessity degree of φ refers to the degree of possibilistic entailment of q.

2) $\alpha = \max\{\beta \in [0,1] \mid \Pi \cup \mathcal{A} \vdash (h,\beta)\}$, *i.e.,* α is the greatest degree of deduction of h from $\Pi \cup \mathcal{A}$;
3) $\mathcal{A}$ is minimal w.r.t. set inclusion, *i.e.,* there is no $\mathcal{A}_1 \subset \mathcal{A}$ such that $\Pi \cup \mathcal{A}_1 \vdash (h,\alpha)$.

It must be remarked that the three conditions in the above definition are inherited from similar definitions in argumentation literature [16]. Moreover, notice that from the above definition of argument, on the basis of a P-DeLP program $\mathcal{P}$, there may exist *different* arguments $\langle \mathcal{A}_1, h, \alpha_1\rangle$, $\langle \mathcal{A}_2, h, \alpha_2\rangle$, ..., $\langle \mathcal{A}_k, h, \alpha_k\rangle$ supporting a given conclusion h, with (possibly) different necessity degrees $\alpha_1, \alpha_2, \ldots, \alpha_k$. For instance, from the program presented in Figure 3 when can build arguments $\langle \mathcal{A}_1, h, 0.39\rangle$ and $\langle \mathcal{A}_2, h, 0.58\rangle$ supporting $h =\sim reconsider$:

$$\mathcal{A}_1 = \left\{ \begin{array}{l} (\sim reconsider \leftarrow high(scoreIH), 0.49), \\ (high(scoreIH), 0.39) \end{array} \right\}$$

$$\mathcal{A}_2 = \left\{ \begin{array}{l} (\sim reconsider \leftarrow high(timeoutIH), 0.66), \\ (high(timeoutIH), 0.58) \end{array} \right\}$$

As indicated above, the *GMP* inference rule allows to propagate necessity degrees; however, given different arguments supporting the same conclusion it is not possible to accumulate their strength in terms of possibilistic values. To do this, in [13] it was defined the notion of *accrued structure* which is recalled in Definition 3. An accrued structure accounts for several arguments supporting the same conclusion and whose necessity degree is defined in terms of two mutually recursive functions: $f^{+}_{\Phi}(\cdot)$ (the accruing function) and $f^{MP}_{\Phi}(\cdot)$ (which propagates necessity degrees as *GMP*).

Given that necessity degrees associated with accrued structures will be used to determine which attack constitutes a defeat, in order to avoid getting committed to a specific way of aggregating necessity degrees, function $f^{+}_{\Phi}(\cdot)$ is parametrized w.r.t. a user-specified function ACC, which must be defined according to the application domain. Additionally, two properties were identified as reasonable to hold for any candidate instantiation of ACC:

Non-depreciation: $ACC(\alpha_1, \ldots, \alpha_n) \geq max(\alpha_1, \ldots, \alpha_n)$ (*i.e.,* accruing arguments results in a necessity degree not lower than any single argument involved in the accrual).

Maximality: $ACC(\alpha_1, \ldots, \alpha_n) = 1$ only if $\alpha_i = 1$ for some i, $1 \leq i \leq n$ (*i.e.,* accrual means total certainty only if there is an argument with necessity degree 1).

Definition 3: Let $\mathcal{P}$ be a P-DeLP program, and Ω be a set of arguments in $\mathcal{P}$ supporting the same conclusion h, *i.e.,* $\Omega = \{\langle \mathcal{A}_1, h, \alpha_1\rangle, ..., \langle \mathcal{A}_n, h, \alpha_n\rangle\}$. The accrued structure for h (or just a-structure) from the set Ω (denoted $Accrual(\Omega)$) is defined as a 3-uple $[\Phi, h, \alpha]$, where $\Phi = \mathcal{A}_1 \cup ... \cup \mathcal{A}_n$ and α is obtained using two mutually recursive functions, $f^{+}_{\Phi}(\cdot)$ and $f^{MP}_{\Phi}(\cdot)$. Let q be a literal appearing in Φ and $(\varphi_1, \beta_1), ..., (\varphi_n, \beta_n)$ be all the weighted clauses in Φ with head q. Then,

$$f^{+}_{\Phi}(q) =_{def} ACC(f^{MP}_{\Phi}(\varphi_1), ..., f^{MP}_{\Phi}(\varphi_n))$$

Let (φ_i, β_i) be a weighted clause in Φ. Then,

$$f^{MP}_{\Phi}(\varphi_i) =_{def} \left\{ \begin{array}{l} \beta_i \\ if\ \varphi_i \text{ is a fact } q; \\ \\ min(f^{+}_{\Phi}(p_1), ..., f^{+}_{\Phi}(p_n), \beta_i) \\ if\ \varphi_i = q \leftarrow p_1, ..., p_n \end{array} \right.$$

Finally, $\alpha = f^{+}_{\Phi}(h)$. When $\Omega = \emptyset$ we get the special accrued structure $[\emptyset, \epsilon, 0]$, representing the accrual of no argument.

Given an a-structure $[\Phi, h, \alpha]$, the set of *arguments in* $[\Phi, h, \alpha]$ is denoted as $Args([\Phi, h, \alpha])$ and it contains all arguments $\langle \mathcal{A}_i, h, \alpha_i\rangle$ s.t. $\mathcal{A}_i \subseteq \Phi$. It is worth noting that $Args([\emptyset, \epsilon, 0]) = \emptyset$.

Hence, the above-mentioned arguments $\langle \mathcal{A}_1, h, 0.39\rangle$ and $\langle \mathcal{A}_2, h, 0.58\rangle$ that support $h =\sim reconsider$ can be accrued to get $[\Phi_1, h, \alpha]$ where $\Phi_1 = \mathcal{A}_1 \cup \mathcal{A}_2$ and in order to compute α, function ACC should be defined. In Section IV-B a concrete ACC is used to compute the α values from the accrued structures obtained from our running example of the TILEWORLD domain. Thus, in the rest of this section, when it is not explicitly needed, we will continue referencing generic aggregated values for accrued structures. Below, three definitions concerning a-structures' properties are introduced.

Definition 4: Let $\mathcal{P}$ be a P-DeLP program. An a-structure $[\Phi, h, \alpha]$ is *maximal* iff $Args([\Phi, h, \alpha])$ contains all arguments in $\mathcal{P}$ with conclusion h.

From Definition 4, we can see that the a-structure $[\Phi_1, h, \alpha]$ referred above is not maximal since there are other arguments like $\langle \mathcal{A}_3, h, 0.03\rangle$, which can also be built from program $\mathcal{P}$ and they do not belong to $Args([\Phi_1, h, \alpha]) = \{\langle \mathcal{A}_1, h, 0.39\rangle, \langle \mathcal{A}_2, h, 0.58\rangle\}$:

$$\mathcal{A}_3 = \left\{ \begin{array}{l} (\sim reconsider \leftarrow low(distAgentIH), 0.16), \\ (low(distAgentIH), 0.03) \end{array} \right\}$$

Definition 5: Let $[\Phi, h, \alpha]$ and $[\Theta, h, \beta]$ be two a-structures. We say that $[\Theta, h, \beta]$ is a *narrowing* of $[\Phi, h, \alpha]$ iff $Args([\Theta, h, \beta]) \subseteq Args([\Phi, h, \alpha])$.

Definition 6: Let $[\Phi, h, \alpha]$ and $[\Theta, k, \gamma]$ be two a-structures. Then we say that $[\Theta, k, \gamma]$ is an *accrued sub-structure* (or just a-substructure) of $[\Phi, h, \alpha]$ iff $\Theta \subseteq \Phi$.

In this way, based on arguments $\mathcal{A}_1$ and $\mathcal{A}_2$ mentioned above, we can obtain two a-structures $[\Theta_1, h, \beta_1]$ and $[\Theta_2, h, \beta_2]$ such that $\Theta_1 = \mathcal{A}_1$ and $\Theta_2 = \mathcal{A}_2$ and it holds that $Args([\Theta_1, h, \beta_1]) \subseteq Args([\Phi_1, h, \alpha])$ and $Args([\Theta_2, h, \beta_2]) \subseteq Args([\Phi_1, h, \alpha])$. That is, both Θ_1 and Θ_2 are narrowings of Φ_1. It is also the case that $[\Theta_1, h, \beta_1]$ and $[\Theta_2, h, \beta_2]$ are a-substructures of $[\Phi_1, h, \alpha]$ since $\Theta_1 \subseteq \Phi_1$ and $\Theta_2 \subseteq \Phi_1$ hold.

2) Modelling Conflict and Defeat among Accrued Structures: An a-structure $[\Phi, h, \alpha]$ stands for (possibly) several chains of reasoning (arguments) supporting the conclusion

h, where some intermediate conclusions in $[\Phi, h, \alpha]$ could be shared by some, but not necessarily all the arguments in $[\Phi, h, \alpha]$. Thus, given two a-structures $[\Phi, h, \alpha]$ and $[\Psi, k, \beta]$, if the conclusion k of $[\Psi, k, \beta]$ contradicts some intermediate conclusion h' in $[\Phi, h, \alpha]$, then only those arguments in $Args([\Phi, h, \alpha])$ involving h' will be affected by the conflict. Next, the notion of *partial attack* is defined, where the attacking a-structure generally affects only a narrowing of the attacked one (that one containing exactly the arguments in the attacked a-structure affected by the conflict). This narrowing will be referred to as the *attacked narrowing*.

Definition 7: Let $[\Phi, h, \alpha]$ and $[\Psi, k, \beta]$ be two a-structures. $[\Psi, k, \beta]$ *partially attacks* $[\Phi, h, \alpha]$ at literal h', iff there exists a complete[5] a-substructure $[\Phi', h', \alpha']$ of $[\Phi, h, \alpha]$ such that $k = \overline{h'}$.[6] The a-substructure $[\Phi', h', \alpha']$ will be called the *disagreement a-substructure.* $[\Lambda, h, \gamma]$ is the *attacked narrowing* of $[\Phi, h, \alpha]$ associated with the attack iff $[\Lambda, h, \gamma]$ is the minimal narrowing of $[\Phi, h, \alpha]$ which has $[\Phi', h', \alpha']$ as an a-substructure.

In the methodology proposed in Section IV-B, there will be no partial attacks to a-substructures, but the notions of *partial attack* and *attacked narrowing* are introduced because they are used in Definition 8. With explanatory purposes, Figure 4 depicts a partial attack from an example taken from [13].

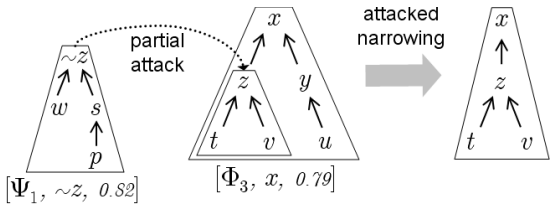

Fig. 4: Partial Attack.

Definition 8: Let $[\Phi, h, \alpha]$ and $[\Psi, k, \beta]$ be two a-structures. Then $[\Psi, k, \beta]$ is a *partial defeater* of $[\Phi, h, \alpha]$ (or equivalently that $[\Psi, k, \beta]$ is a successful attack on $[\Phi, h, \alpha]$) iff (1) $[\Psi, k, \beta]$ attacks $[\Phi, h, \alpha]$ at literal h', where $[\Phi', h', \alpha']$ is the disagreement a-substructure, and (2) $\beta \geq \alpha'$.

In order to exemplify the notion of partial defeater, we need an a-structure such that is in disagreement with those mentioned above, viz. $[\Theta_1, h, \beta_1]$, $[\Theta_2, h, \beta_2]$ and $[\Phi_1, h, \alpha]$. From program $\mathcal{P}$, depicted in Figure 3, only one such a-structure can be built; namely, $[\Phi_2, reconsider, \alpha']$. On the grounds that $Args([\Phi_2, reconsider, \alpha']) = \{\langle \mathcal{A}_4, reconsider, 0.92\rangle\}$, then $\alpha' = 0.92$:

$$\mathcal{A}_4 = \left\{ \begin{array}{l} (reconsider \leftarrow low(holeGestation), 0.99), \\ (low(holeGestation), 0.92) \end{array} \right\}$$

Similarly, since $Args([\Theta_1, h, \beta_1]) = \{\langle \mathcal{A}_1, h, 0.39\rangle\}$ and $Args([\Theta_2, h, \beta_2]) = \{\langle \mathcal{A}_2, h, 0.58\rangle\}$, then $\beta_1 = 0.39$ and $\beta_2 = 0.58$, respectively; with $h = \sim reconsider$. In this way, from Definition 8, we can state that $[\Phi_2, reconsider, 0.92]$ is a partial defeater for a-structures $[\Theta_1, h, 0.39]$ and $[\Theta_2, h, 0.58]$ since $0.92 > 0.39$ and $0.92 > 0.58$ hold, being the disagreement a-substructures the a-structures themselves.

Definition 9: Let $[\Phi, h, \alpha]$ and $[\Psi, k, \beta]$ be two a-structures such that $[\Psi, k, \beta]$ attacks $[\Phi, h, \alpha]$. Let $[\Lambda, h, \gamma]$ be the attacked narrowing of $[\Phi, h, \alpha]$. Then the *defeated narrowing* of $[\Phi, h, \alpha]$ associated with the attack, denoted as $\mathsf{N}^{\mathsf{D}}_{\mathsf{w}}([\Phi, h, \alpha], [\Psi, k, \beta])$, is defined by cases as follows:

1) $\mathsf{N}^{\mathsf{D}}_{\mathsf{w}}([\Phi, h, \alpha], [\Psi, k, \beta]) =_{def} [\Lambda, h, \gamma]$, if $[\Psi, k, \beta]$ is a partial defeater of $[\Phi, h, \alpha]$, or
2) $\mathsf{N}^{\mathsf{D}}_{\mathsf{w}}([\Phi, h, \alpha], [\Psi, k, \beta]) =_{def} [\emptyset, \epsilon, 0]$, otherwise.

Similarly, with $\mathsf{N}^{\mathsf{U}}_{\mathsf{w}}([\Phi, h, \alpha], [\Psi, k, \beta])$, we denote the undefeated narrowing of $[\Phi, h, \alpha]$ associated with the attack, and corresponds to the accrued structure $Accrual(Args([\Phi, h, \alpha]) \backslash Args(\mathsf{N}^{\mathsf{D}}_{\mathsf{w}}([\Phi, h, \alpha], [\Psi, k, \beta])))$.

Given an attack relation, we will identify two complementary narrowings associated with the attacked a-structure: the narrowing that becomes defeated as a consequence of the attack, and the narrowing that remains undefeated. To exemplify these concepts we will resort to an extension of the generic example posed in [13] and that was previously introduced in Figure 4. Accordingly, Figure 5 illustrates a successful attack from $[\Psi_1, \sim z, 0.82]$ against $[\Phi_3, x, 0.79]$, as well as the associated defeated and undefeated narrowings of $[\Phi_3, x, 0.79]$.

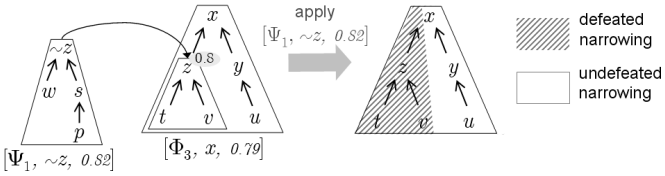

Fig. 5: Defeated and Undefeated Narrowings.

3) Dialectical Analysis for Accrued Structures: Given a program $\mathcal{P}$ and a literal h, we are interested in determining if h is ultimately accepted (or *warranted*), and if so, with which necessity degree. In order to determine this, the maximal a-structure $[\Phi, h, \alpha]$ supporting h will be considered, and the final undefeated narrowing of $[\Phi, h, \alpha]$ after considering all the possible a-structures attacking it, will also be analysed.

As those attacking a-structures may also have other a-structures attacking them, this strategy prompts a recursive dialectical analysis formalized by the concept of accrued dialectical tree, denoted as $\mathcal{T}_h$.

In the implementation proposed in Section IV-B, each dialectical tree will consist of only one node, the root that will be also a leaf. Therefore, it is not necessary to introduce definitions related with this concept in order to maintain the theoretical background as reduced as possible. The interested reader is encouraged to see [13] for a proper formalization.

Definition 10: Let $\mathcal{P}$ be a P-DeLP program and let h be a literal. Let $[\Phi, h, \alpha]$ be the maximal a-structure for h such that its undefeated narrowing in $\mathcal{T}_h$ is a non-empty a-structure $[\Phi', h, \alpha']$. Then we say that h is *warranted* w.r.t. $\mathcal{P}$ with necessity α' and that $[\Phi', h, \alpha']$ is a *warranted a-structure*.

[5]We say that $[\Phi', h', \alpha']$ is a complete a-substructure of $[\Phi, h, \alpha]$, iff for any other a-substructure $[\Phi'', h'', \alpha'']$ of $[\Phi, h, \alpha]$, it holds that $\Phi'' \subset \Phi'$.

[6]For a given literal h, we will write $\overline{h}$ to denote "$\sim a$" if $h \equiv a$, and "a" if $h \equiv \sim a$.

An example of warranted a-structure will be directly introduced in the following section, in the context of the implementation proposed for IR as an argumentative process that mades a dichotomous choice.

B. A P-DeLP implementation

As it can be observed from Algorithm 3, given a P-DeLP program with certain features (line 1), the $select(\cdot,\cdot,\cdot)$ function returns "true" if the literal $reconsider$ is warranted from $\mathcal{P}$ and "false" in another case. So, from a conceptual point of view, the argumentative approach is easy to understand. Below, in Section IV-B1, we explain the technical details involved in developing this particular P-DeLP program, so-called *conformable* program, that allows solving this decision problem by using accrual of arguments [13].

As mentioned above, in order to decide whether to reconsider or not, we consider a criteria set $C = \{C_1, ..., C_n\}$ related to the intended hole (IH) and the environment where the agent operates. For each criterion $C_i \in C$, a range $[\min_i, \max_i]$ is provided, representing the values $v(C_i)$ which are expected for this evaluation dimension. Besides, we also have the thresholds set $\Theta = \{\theta_1, ..., \theta_n\}$; that is, for each criterion $C_i \in C$ we have a threshold θ_i for determining whether its value is high ($v(C_i) \geq \theta_i$) or low ($v(C_i) < \theta_i$). Finally, in order to obtain the preference set $W = \{w_1, ..., w_n\}$, where $w_i \in [0, 1]$ represents how important is criterion C_i in the decision making process, a relationship of strict total order on set C should be given. Thus, criteria must be sorted from lowest to highest preference and a ranking from 1 to n should be obtained. Then, $w_i = (r/n) - 0.01$ where r is the position that C_i occupies in the ranking. The value 0.01 is subtracted because the highest preference criterion which could result is $w = 1$, which stands for a certain clause in a P-DeLP program, and given the decision approach that we are following [17], certain clauses are not considered to develop the conformable P-DeLP program referred in line 1 of Algorithm 3.

Algorithm 3 Computation for alternatives selection (P-DeLP with argument accrual)

function *select*(alternatives A, criteria C, preferences $\mathcal{P}$) **returns** boolean
//$A = \{yes, no\}$
//$C = \{\langle\ C_1, \theta_1, \min_1, \max_1\rangle, ..., \langle\ C_n, \theta_n, \min_n, \max_n\rangle\}$
//$\mathcal{P} = w = \{w1, ..., w_n\}$

1: Build a conformable P-DeLP program $\mathcal{P}$ as specified in Section IV-B1.
2: Determine if literal $h = reconsider$ is ultimately accepted (or warranted).
3: **if** h is warranted **then**
4: **return** true
5: **else**
6: **return** false
7: **end if**

In the context of the running example we are using, Table II shows all the values mentioned above for each criterion C_i.

TABLE II: Values used to compute Intention Reconsideration through Argumentation.

C	r	$\min_i$	$\max_i$	w_i	$v(C_i)$	θ_i	φ	α'_i	α_i
C_1	6	1	60	0.99	5	10	low	0.55	0.92
C_2	5	0	100	0.82	100	50	$high$	1	0.82
C_3	2	0	100	0.32	100	30	$high$	1	0.32
C_4	3	1	10	0.49	6	3	$high$	0.43	0.39
C_5	4	1	20	0.66	14	7	$high$	0.54	0.58
C_6	1	0	11	0.16	4	5	low	0.2	0.03

1) Building a conformable P-DeLP program: The above-mentioned conformable P-DeLP program $\mathcal{P}$ is made following the three steps detailed below:

1: For each criterion C_i a clause (φ, α) must be added. If $v(C_i) \geq \theta_i$, then the literal φ is $high(C_i)$ and $\alpha' = (v(C_i) - \theta_i)/(\max_i - \theta_i)$. On the contrary, when $v(C_i) < \theta_i$ holds, φ is $low(C_i)$ and $\alpha' = (\theta_i - v(C_i))/(\theta_i - \min_i)$. The value α' represents how high (or low) is the value $v(C_i)$ with respect to threshold θ_i. Finally, the necessity degree α is obtained as $\alpha = \frac{\alpha'-1}{n} + w_i$, which ponders the value α' based on the weight w_i of criterion C_i.

Considering criterion $C_1 = holeGestation$ as example, and the information given by the first row of Table II, the clause $(low(holeGestation), 0.92)$ is included into the program $\mathcal{P}$. The clause is of the type $low(C_i)$ given that $v(C_1) = 5$ is lower than the threshold $\theta_1 = 10$. The necessity degree α_1 is obtained as $\alpha_1 = \frac{\alpha'-1}{6} + 0.99 = 0.92$ where $\alpha' = (10 - 5)/(10 - 1) = 0,55$. Below, all the clauses that would be added to program $\mathcal{P}$ in this stage are shown:

$$\left\{\begin{array}{l}(low(holeGestation), 0.92)\\(low(distAgentIH), 0.03)\\(high(deterministic), 0.82)\\(high(accesibility), 0.32)\\(high(scoreIH), 0.39)\\(high(timeoutIH), 0.58)\end{array}\right\} = \mathcal{P}_1$$

2: A clause $(reconsider \leftarrow \phi, \alpha)$ must be included for each criterion C_i. If it is the case that a value $v(C_i)$ greater than threshold θ_i is a reason in favor of reconsideration, then $\phi = high(C_i)$; else, $\phi = low(C_i)$. The necessity degree α matches w_i, the criterion's weight.

According to the criteria chosen for our running example, the agent should reconsider its intentions when: $C_1 = gestationHole$ is low, $C_2 = deterministic$ is low, $C_3 = accesibility$ is low, $C_4 = scoreIH$ is low, $C_5 = timeoutIH$ is low, and $C_6 = distAgent$ is high. That is:

$$\left\{\begin{array}{l}(reconsider \leftarrow low(holeGestation), 0.99)\\(reconsider \leftarrow low(deterministic), 0.82)\\(reconsider \leftarrow low(accesibility), 0.32)\\(reconsider \leftarrow low(scoreIH), 0.49)\\(reconsider \leftarrow low(timeoutIH), 0.66)\\(reconsider \leftarrow high(distAgentIH), 0.16)\end{array}\right\} = \mathcal{P}_2$$

3: A clause $(\sim reconsider \leftarrow \phi, \alpha)$ must be added for each criterion C_i. If it is the case that a value $v(C_i)$ greater than threshold θ_i is a reason against reconsidering, then $\phi = high(C_i)$; else, $\phi = low(C_i)$. Again, the necessity degree α matches w_i, the criterion's weight.

Returning to our example, the agent should not reconsider its intentions if: $C_1 = gestationHole$ is high, $C_2 = deterministic$ is high, $C_3 = accesibility$ is high, $C_4 = scoreIH$ is high, $C_5 = timeoutIH$ is high, and $C_6 = distAgent$ is low. That is:

$$\left\{ \begin{array}{l} (\sim reconsider \leftarrow high(holeGestation), 0.99) \\ (\sim reconsider \leftarrow high(deterministic), 0.82) \\ (\sim reconsider \leftarrow high(accesibility), 0.32) \\ (\sim reconsider \leftarrow high(scoreIH), 0.49) \\ (\sim reconsider \leftarrow high(timeoutIH), 0.66) \\ (\sim reconsider \leftarrow low(distAgentIH), 0.16) \end{array} \right\} = \mathcal{P}_3$$

In this way, as introduced in Section IV-A, the complete conformable program would be $\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3$, which was formerly introduced in Figure 3. Next, in order to decide whether to reconsider or not, an aggregate function ACC must be provided. In our case, we use the ACC function proposed in [17] that is recalled below:

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

with $k \in (0, 1)$, which has been shown in [17] to satisfy the desired properties about accrual functions stated above in Section IV-A. It is worth mentioning that k aims at weighting the importance given to the highest priority preference criterion. For our running example, we will set $k = 0.1$ as performed in [17]; but naturally, higher values could be used if we were interested in giving more decision power to those criteria ranked as most important.

In this way, as stated in Algorithm 3, with the conformable P-DeLP program built it only remains to check whether the literal $h = reconsider$ can be warranted. According to the criteria chosen in our example, the agent should reconsider its intentions if: $gestationHole$ is low, $deterministic$ is low, $accesibility$ is low, $scoreIH$ is low, $timeoutIH$ is low, and $distAgent$ is high. Conversely, the agent should not reconsider if: $gestationHole$ is high, $deterministic$ is high, $accesibility$ is high, $scoreIH$ is high, $timeoutIH$ is high, and $distAgent$ is low. Hence, the following arguments, $\langle \mathcal{A}_1, h, 0.92\rangle$, $\langle \mathcal{A}_2, \sim h, 0.82\rangle$, $\langle \mathcal{A}_3, \sim h, 0.32\rangle$, $\langle \mathcal{A}_4, \sim h, 0.39\rangle$, $\langle \mathcal{A}_5, \sim h, 0.58\rangle$, $\langle \mathcal{A}_6, \sim h, 0.03\rangle$ are obtained from $\mathcal{P}$, supporting conclusions h and $\sim h$ respectively, with:

$$\mathcal{A}_1 = \left\{ \begin{array}{l} (reconsider \leftarrow low(holeGestation), 0.99), \\ (low(holeGestation), 0.92) \end{array} \right\}$$

$$\mathcal{A}_2 = \left\{ \begin{array}{l} (\sim reconsider \leftarrow high(deterministic), 0.82), \\ (high(deterministic), 0.82) \end{array} \right\}$$

$$\mathcal{A}_3 = \left\{ \begin{array}{l} (\sim reconsider \leftarrow high(accesibility), 0.32), \\ (high(accesibility), 0.32) \end{array} \right\}$$

$$\mathcal{A}_4 = \left\{ \begin{array}{l} (\sim reconsider \leftarrow high(scoreIH), 0.49), \\ (high(scoreIH), 0.39) \end{array} \right\}$$

$$\mathcal{A}_5 = \left\{ \begin{array}{l} (\sim reconsider \leftarrow high(timeoutIH), 0.66), \\ (high(timeoutIH), 0.58) \end{array} \right\}$$

$$\mathcal{A}_6 = \left\{ \begin{array}{l} (\sim reconsider \leftarrow low(distAgentIH), 0.16), \\ (low(distAgentIH), 0.03) \end{array} \right\}$$

It is worth mentioning that arguments $\langle \mathcal{A}_4, \sim h, 0.39\rangle$, $\langle \mathcal{A}_5, \sim h, 0.58\rangle$, $\langle \mathcal{A}_6, \sim h, 0.03\rangle$ and $\langle \mathcal{A}_1, h, 0.92\rangle$, have been previously introduced in Section IV-A with illustrative purposes, as arguments $\langle \mathcal{A}_1, \sim h, 0.39\rangle$, $\langle \mathcal{A}_2, \sim h, 0.58\rangle$, $\langle \mathcal{A}_3, \sim h, 0.03\rangle$ and $\langle \mathcal{A}_1, h, 0.92\rangle$, respectively.

Based on the arguments shown above, the following a-structures $\Phi = \mathcal{A}_1$ and $\Phi' = \mathcal{A}_2 \cup \mathcal{A}_3 \cup \mathcal{A}_4 \cup \mathcal{A}_5 \cup \mathcal{A}_6$ can obtained from the process of arguments accrual:

$$\begin{aligned} &Accrual(\{\langle \mathcal{A}_1, reconsider, 0.92\rangle\}) \\ &\quad = [\Phi, reconsider, 0.92] \end{aligned}$$

$$\begin{aligned} &Accrual(\{\langle \mathcal{A}_2, \sim reconsider, 0.82\rangle, \\ &\qquad \langle \mathcal{A}_3, \sim reconsider, 0.32\rangle, \\ &\qquad \langle \mathcal{A}_4, \sim reconsider, 0.39\rangle, \\ &\qquad \langle \mathcal{A}_5, \sim reconsider, 0.58\rangle, \\ &\qquad \langle \mathcal{A}_6, \sim reconsider, 0.03\rangle\}) \\ &\quad = [\Phi', \sim reconsider, 0.97] \end{aligned}$$

We can see from Definition 4 that a-structures $[\Phi, reconsider, 0.92]$ and $[\Phi', \sim reconsider, 0.97]$ are both maximal. Besides, from Definition 7 we can state that a-structure $[\Phi', \sim reconsider, 0.97]$ attacks a-structure $[\Phi, reconsider, 0.92]$. Moreover, from Definition 8, it can be observed that $[\Phi, reconsider, 0.92]$ itlself is the atacked narrowing of $[\Phi', \sim reconsider, 0.97]$ and we can also draw that $[\Phi', \sim reconsider, 0.97]$ is a successful attack on $[\Phi, reconsider, 0.92]$ since $0.97 > 0.92$ holds. Hence, by Definition 9 we can state that $[\Phi, reconsider, 0.92]$ is the defeated narrowing, and from Definition 10, we conclude that $h = reconsider$ is not warranted and therefore $reconsider(\cdot, \cdot)$ function return false. In other words, the agent believes there are no sufficient reasons for having to reconsider its intentions.

V. Conclusions

One of the features that greater attention must receive regarding the BDI agents is their intention reconsideration strategy. For this reason, in this work, we have presented an intention reconsideration strategy implemented through an argumentative formalism, P-DeLP, which uses accrual of arguments to support warranted conclusions. Similarly to proposals [2] and [4], our proposal agrees about that IR is formalized as a meta-reasoning problem. At each time step,

the agent faces a choice between two dichotomous alternatives, viz. acting or deliberating; and this allows it to choose by itself which strategy to adopt depending on the current state of the environment. In other words, the agent chooses in an autonomous way a level of commitment according to the current state of the environment, as opposed to fixed strategies (cautious and bold agents) presented in [3]. However, our approach not only considers the dynamism of the environment but also it explicitly considers environmental characteristics, such as determinism and accessibility. In addition, our mechanism also considers particular characteristics of the current intention when making a decision.

As future work, we will study the optimality of the proposed reconsideration strategy following the guidelines from [9]. We will also conduct experimental studies on the behavior and effectiveness of our reconsideration strategy in environments with different degrees of dynamism, accessibility and non-determinism. Moreover, an empirical comparison between the aforementioned works and our proposed mechanism is planned.

ACKNOWLEDGMENT

We thank to the three anonymous reviewers for their careful reading of our manuscript and their comments and suggestions. This work has been partially founded by Proyecto PROICO P-31816, Universidad Nacional de San Luis, Argentina.

REFERENCES

[1] M. Wooldridge, *Reasoning about Rational Agents*. The MIT Press, 2000.

[2] M. Schut and M. Wooldridge, "Intention reconsideration in complex environments," in *4th International Conference on Autonomous Agents*, 2000.

[3] D. N. Kinny, "Commitment and effectiveness of situated agents," in *12th International Joint Conference on Artificial Intelligence (IJCAI)*, 1991, pp. 82–88.

[4] M. van Zee and T. Icard, "Intention reconsideration as metareasoning," in *Bounded Optimality and Rational Metareasoning NIPS 2015 Workshop*, December 2015.

[5] C. S. Toranzo, M. Errecalde, and E. Ferretti, "On the use of agreement technologies for multi-criteria decision making within a BDI agent," in *Advances in Artificial Intelligence - IBERAMIA 2014, Proceedings*, ser. LNCS, vol. 8864. Springer, 2014, pp. 54–65.

[6] C. Sosa-Toranzo, M. Errecalde, and E. Ferretti, "Intention reconsideration like uncertain dichotomous choice model," in *XX Congreso Argentino de Ciencias de la Computación (CACIC)*, October 2014, pp. 1248–1257.

[7] M. E. Pollack and M. Ringuette, "Introducing the tileworld: Experimentally evaluating agent architectures," in *8th AAAI*, 1990, pp. 183–189.

[8] M. Bratman, D. Israel, and M. Pollack, "Plans and resource bounded reasoning," *Computational Intelligence*, vol. 4, no. 4, pp. 349–355, 1988.

[9] M. Wooldridge and S. Parsons, "Intention reconsideration reconsidered," in *Intelligent Agents V: Agents Theories, Architectures, and Languages*, ser. LNCS. Springer, 1999, vol. 1555, pp. 63–79.

[10] C. Sosa-Toranzo, M. Errecalde, and E. Ferretti, "A framework for multi-criteria argumentation-based decision making within a BDI agent," *JCS&T*, vol. 14, no. 1, 2014.

[11] C. Chesñevar, G. Simari, T. Alsinet, and L. Godo, "A logic programming framework for possibilistic argumentation with vague knowledge," in *20th conference on Uncertainty in Artificial Intelligence (UAI)*, Banff, Canada, 2004, pp. 76–84.

[12] T. Alsinet, C. I. Chesñevar, L. Godo, and G. Simari, "A logic programming framework for possibilistic argumentation: formalization and logical properties," *Fuzzy Sets and Systems*, vol. 159, no. 10, 2008.

[13] M. Gómez, C. Chesñevar, and G. Simari, "Modelling argument accrual in possibilistic defeasible logic programming," in *ECSQARU*, ser. LNCS, vol. 5590. Springer, 2009, pp. 131–143.

[14] V. Lifschitz, "Foundations of logic programming," in *Principles of Knowledge Representation*, G. Brewka, Ed. Stanford, California: CSLI Publications, 1996, pp. 69–127.

[15] A. García and G. Simari, "Defeasible logic programming: an argumentative approach," *Theory and Practice of Logic Programming*, vol. 4, no. 2, pp. 95–138, 2004.

[16] C. Chesñevar, A. Maguitman, and R. P. Loui, "Logical models of argument," *ACM Computing Surveys*, vol. 32, no. 4, pp. 337–383, 2000.

[17] E. Ferretti, M. L. Errecalde, A. J. García, and G. R. Simari, "A possibilistic defeasible logic programming approach to argumentation-based decision-making," *Journal of Experimental & Theoretical Artificial Intelligence*, June 2014.