An Architecture for Argumentation-based Epistemic Planning: A First Approach with Contextual Preferences

Juan Carlos Teze[1,2] and Lluis Godo[3]
[1]Grupo de Investigación en Agentes y Sistemas Inteligentes (GINAySI),
Laboratorio de Informática y Sistemas, Facultad de Ciencias de la Administración,
Universidad Nacional de Entre Ríos, (E3202BHR) Concordia, Entre Ríos, Argentina
[2]Institute for Computer Science and Engineering (UNS–CONICET),
Universidad Nacional del Sur (UNS) & Consejo Nacional de Investigaciones Cientificas y Tecnicas (CONICET),
San Andres 800, Campus Palihue, (8000) Bahia Blanca, Argentina
[3]Artificial Intelligence Research Institute (IIIA-CSIC)
Campus UAB - 08193 Bellaterra, Barcelona, Spain
Corresponding author: Juan Carlos Teze (email: carlos.teze@uner.edu.ar).

***Abstract* — Argumentation-based planning systems present an interesting proposal for complex and dynamic domains where defeasible argumentation is used in the reasoning processes used during the construction of plans by considering the available knowledge. In many real-world application scenarios, knowledge is provided with explicit priorities. Despite its importance, existing planning systems do not provide additional reasoning capacities of dynamically changing the preferences expressed by these priorities when a plan is being constructed. In this work, we present an argumentation and planning architecture, and propose a set of software engineering guidelines to analyze and design planning systems leveraging this capacity.**

***Index Terms*—Planning, Contextual Preference, Defeasible Argumentation.**

I. Introduction

The introduction of epistemic elements when building plans to obtain a given goal has revealed an exciting and useful new perspective in the planning research area. This has been succinctly described by T. Bolander in [1]: "*Epistemic planning is the enrichment of planning with epistemic notions, that is, knowledge and beliefs.*" Defeasible argumentation is a form of commonsense reasoning that agents can use to exploit their knowledge bases (KBs), which are possibly inconsistent [2]. Defeasible argumentation-based epistemic planners (DAEPs, for short) have shown their potential in complex and dynamic domains where unresolved contradictory information situations and incomplete information are present [3], and they are characterized by the efficient use of defeasible reasoning for all the epistemic tasks performed over the represented knowledge. Specifically, the fundamental process in defeasible argumentation is to confront reasons to support or dismiss a conclusion that is under scrutiny. An analysis mechanism supports this process by obtaining arguments and then comparing those in conflict to decide possible defeaters; this last step requires a comparison, which in turn needs a preference criterion defined on the set of arguments to reach an acceptance decision. This analysis has a valuable additional result: the inference mechanism can be used to reason about the preconditions and effects of a planner agent's actions.

DAEPs traditionally adopt a single perspective, which is reflected by the preference criterion used to establish why an argument is favored over others; therefore, these agents should be equipped with an appropriate preference criterion for the domain where they are deployed. Given the usual dynamic nature of preferences, to design a real planner agent with a single preference criterion when it has to face different situations is quite limiting; besides, when the comparison is indecisive, an agent could become ineffective. This situation improves when this type of planner expands its capacities to consider contextual preferences [4], for instance in terms of defeasible rules with weights expressing a preference strength that may vary from a particular context to another. A possible way to address such a circumstance is to consider tools that allow dynamically changing these preferences according to the current context as the agent reasons to select which actions to add to a plan. In our proposal, instead of considering only a priority order at knowledge level under a single preference context, the agent can evaluate the situation considering all available reasoning contexts and decide which one to use depending on the current state of the world. The use of a conditional preference selection mechanism that considers the context allows agents to adopt the context that better adapts to the agent's knowledge about the situation in which the reasoning is performed.

Several works have proposed using argumentation to enhance planning systems. Defeasible planning was arguably triggered by Pollock' proposal [5] with its planner OSCAR to reason defeasibly about plans as a whole, in the sense of inferring defeasibly whether a plan is a solution of a planning problem. On the other hand, early work such as García et al. [3] stressed the importance of considering epistemic elements during the construction of plans. Along the same line, more recently, Teze et al. [6] presented an application to assist in policy-making tasks in the domain of agricultural water use. Pajares-Ferrando and Onaindia [7], [8] introduced argumentation-based tools in multi-agent planning, with applications in ambient intelligence. In [9] Pardo and Godo proposed a dialogue-based approach to multi-agent collaborative planning based on a temporal defeasible argumentation.

Finally, Oren et al. [10] presented a tool for explaining plans that make use of formal argumentation and dialogue-theory. Nevertheless, it is interesting to remark that these efforts did not focus on the specification of preferences. The work presented here describes a planning framework and a set of guidelines to support knowledge and software engineers in the analysis and design of argumentation-based planning systems with the distinguishing capability of handling preferences, aiming to fill this gap in the current intelligent systems development literature.

II. Analysis and Design of Planning Systems with Defeasible Reasoning and Preferences

When analyzing and designing planning systems with these characteristics, it is desirable to focus on five central aspects: 1) the analysis of the planning domain and the representation of preferences; 2) the description of the planning problem and the associated preferences; 3) the planning mechanism; 4) the reasoning mechanism, and 5) the design of user interactions. We now introduce our architecture (see Figure 1), along with a series of methodological guidelines designed to address these aspects.

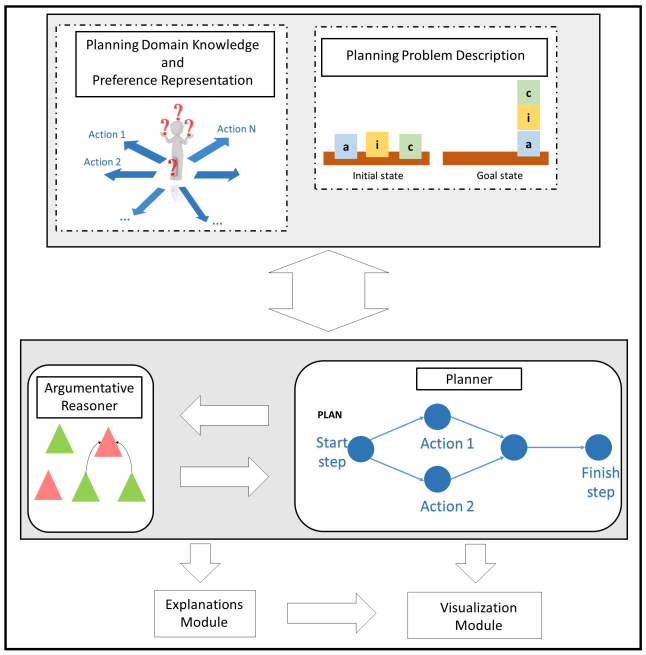

Fig. 1: The Epistemic Planning Framework based on Argumentation.

1) Planning Domain Analysis and Preference Representation

In solving a planning problem, planning systems should be provided with an appropriate set of actions to perform. The representation of these actions must consider all the preconditions and effects that are relevant to solve the problem. In many real-world applications, these systems often have potentially contradictory and incomplete knowledge about the environment. In this context, structured argumentation has played a significant role in capturing and representing this type of knowledge to be used in reasoning about actions. Particularly, the KB is the structure where knowledge of the domain is formally represented.

This stage's result is a detailed and precise description of the planning domain and the user's preferences. The generation of a KB can be carried out in four steps:

- (i) Analyzing the domain and characterizing the set of states of the environment where the system will act, together with the set of actions that indicate the transitions between these states.
- (ii) Specifying a preference relation among information pieces.
- (iii) Specifying an argument preference criterion to apply in case the rules established in the previous steps generate contradictory results.
- (iv) Describing the KB in a formal logical language.

We create statements in natural language for the encoding of different forms of knowledge expressing domain information during the first step. It is necessary to appeal to domain experts to generate rules, and in the second and third steps, elicit the user's preferences.

Preference criteria are established to compare information and deciding between conflicting arguments. The argument comparison criterion must reflect the domain's salient characteristics, since if this criterion fails to do that, the system could often fail in its primary role of making a decision. Different criteria are possible for computing a preference relation over arguments. These preferences usually reflect the importance or priority of the information that arguments contain. Typically, we can encounter two types of preferences or priorities [11]:

- Implicit, that are extracted from the KB due to the defeasible nature of the information it contains. For example, in the case the KB consists of a set of defeasible rules, arguments can be ordered by exploiting specificity relations between their rules [12].
- Explicit, that are specified by stratifying the KB, for instance by attaching weights to each piece of information in the KB representing certainty or perceived priority.

Finally, in the fourth step, the KB should be specified in a formal language to be interpreted by the reasoner. In contrast to argumentation-based planning, traditional planning methods cannot be applied directly in this final step since their descriptions assume a fully observable, static, and deterministic world where the available knowledge might be incomplete or inconsistent, a situation which might lead to contradictory conclusions.

2) Planning Problem Analysis

Classical planning aims at finding a sequence of actions that, starting from an initial state, leads to a goal state. However, it is often the case that certain approaches are concerned not only with the final goal state after plan execution, but also they attempt to address other important aspects, such as user's preferences [13], value-driven actions [6], and norms imposed on the system establishing what the system is required to do under certain conditions [14]. In particular, our proposal addresses the novel question of the representation and application of contextual preferences in each available plan action. Here, we focus on defining a way of modifying preferences, through the

weights attached to defeasible rules, depending on contextual information at the moment of evaluating the preconditions of an action.

3) Analysis and Design of the Reasoner

The argumentative reasoner is the system's main component, which interprets available knowledge, obtains arguments, and then compares those in conflict to decide on acceptance. A reasoner contains three main components:

- *Inference Mechanism*, which analyzes domain knowledge inferring new knowledge to be used during the construction of plans.
- *Conflict Solver*, which evaluates the relationships between arguments, using a preference relation established over the set of arguments through a preference criterion. The knowledge engineer generally specifies this criterion, which should be strongly related to the application domain.
- *Semantic Analyzer*, which aims at determining the acceptability of arguments by considering the interaction between them. Given an argument, it considers its defeaters, the defeats of its defeaters, and so on, in an exhaustively recursive process. Essentially, there are two forms for doing this: declaratively and procedurally. The former establishes conditions that a set of acceptable arguments must meet [15], while the latter provides a specific algorithm [12].

4) Analysis and Design of the Planner

The planner controls and articulates interactions among the components mentioned above, and it is in charge of obtaining a sequence of actions to achieve the desired goals making use of defeasible reasoning in this process. Most of the proposals in the literature generally consider one of the following two approaches: either the whole plan is viewed as an argument, and then defeasible reasoning is performed over complete plans, or it is used as a tool for determining which actions are applicable in a given state; in both cases, defeasible reasoning is performed to select actions.

Planning algorithms are mainly based on two approaches: progression planning and regression planning. A progression planner searches forward from a given initial state until a goal state is reached. Naturally, if there is a considerable number of actions, then the branching factor could be very large, and the search problem becomes intractable. A regression planner tries to improve this situation by beginning from the goal state and generating the plan in inverse order.

5) Design of Outputs

Once all the modules for constructing plans are in place, the next step is to design their presentation to users; this consists in designing the system interface and any mechanism to translate them in an appropriate way to be presented to the user. In turn, this step can include effective ways to interpret the process by which the planning decisions are made. Plan explainability is crucial for helping users to understand plans. Visual plan explanation [16] presents a simple graphical view of a plan, with nodes representing actions, edges linking them, and different filtering options available to the user. Other approaches [17] involve a textual description of the plan in natural language. Another related work [10] uses argumentation for explanations, providing mechanisms to construct arguments that can be useful to justify why a plan should be executed, note aspects of a plan, and exploit the use of causality for explanations.

III. Case study: P-DeLP-based instantiation

Possibilistic Defeasible Logic Programming (P-DeLP) [18] is a structured argumentation framework that extends the DeLP framework [12] by allowing to attach priority or preference weights to defeasible rules. This section will sketch a P-DeLP-based particular instantiation of the proposed architecture, as represented in Fig. 1. We only present a limited version of the analysis carried out in each stage for reasons of space, focusing on showing the intermediate results towards obtaining the final planning system. We will also show how this instantiation can be applied to a shopping planning service.

The application domain consists of a scenario where a shopping agent must find IT product offers considering users' preferences and domain information, *e.g.,* products in stock. The agent offers a product once the shopping website has been chosen and the product selected. The agent task finalizes with the selected products in a virtual shopping cart. Our approach expands a planner's capacity by considering contextual preferences, *i.e.,* preferences over defeasible knowledge expressed in a particular context.

We consider now a KB representing the domain (defeasible) knowledge and preferences among different pieces of defeasible knowledge formalized as a program in P-DeLP. We will summarize the elements we require here from the works in [18].

A *weighted* clause is a pair $(R;\omega)$, where R is a rule $L \leftarrow L_1, \ldots, L_k$ or a fact L (*i.e.,* a rule with empty antecedent), $L, L_1, \ldots, L_k$ are literals, and the weight $\omega \in (0, 1]$ expresses the priority or preference level of the rule, mathematically interpreted as a lower bound for the necessity degree of R. What weights allow is a ranking of rules in terms of their priority level. Following the usual notation of Logic Programming [19], we will regard the set of literals in the body of a clause $L_1, \ldots, L_k$ as a conjunction of these literals. As in [12], P-DeLP rules can also be represented as schematic rules with variables; schematic variables are denoted with an initial uppercase letter. A clause $(R;\omega)$ is referred as *certain* or *strict* when $\omega = 1$, and *uncertain* or *defeasible* when $\omega < 1$. Two components describe a set $\mathbb{P}$ of weighted clauses–often referred to as a P-DeLP-program or simply a *program*, the set of all the clauses in $\mathbb{P}$ considered as strict, denoted Π, and the set of all the defeasible clauses in $\mathbb{P}$, denoted Δ. When useful, we will write $\mathbb{P} = (\Pi, \Delta)$ to refer to the set of weighted clauses, discriminating certain and uncertain clauses.

When reasoning with contradictory and dynamic information, the P-DeLP-system builds arguments from a program $\mathbb{P}$. An argument $\mathcal{A}$ for a literal L is a minimal, non-contradictory set of defeasible rules such that together with the program's strict knowledge allows the derivation of L with a given weight ω (the smallest weight of the clauses involved in the derivation), that will be regarded as the conclusion supported

by $\mathcal{A}$. The ultimate answer to queries will be based on the existence of warranted arguments, computed through dialectical analysis. In P-DeLP, a literal L is warranted from a given program if there exists an argument $\mathcal{A}$ supporting L that cannot be defeated (see [18] for more details about the warrant process).

Prioritized information is particularly useful to select appropriate knowledge and guide planning process according to the user needs. Despite its importance, existing planning systems do not provide additional reasoning capacities of dynamically changing the preferences expressed by these priorities when a plan is being constructed. Given this consideration, our proposal provides planning agents with the capability of changing how they decide their preferences through the use of context-adaptable mechanisms.

In particular, we propose here a context-dependent approach for the assignment of weights to defeasible rules. As already mentioned, these assignments are meant to express a form of preference or priority as defined e.g. in [20]. Here the notion of context is understood in a general sense as conditions favouring a particular preference criterion. Namely, given a program, the set of defeasible rules prioritized according to a particular criterion crit will be denoted as a *priority context*, which can be specified as an assignment $\mathsf{prOrder}_{\mathsf{crit}}$ of weights to defeasible rules. In the following, we assume that a planner will have a set of priority contexts specifically defined for the planning problem represented knowledge.

In our proposal, the state of the world is represented by the epistemic planning agent as a consistent set of weighted literals. From the shopping scenario sketched before, we consider the following state of the world:

$$\Psi = \left\{ \begin{array}{l} (client(juan, ncbahia);\ 1) \\ (trust(maria, juan);\ 1) \\ (goodPrice(sony\_hph, ncbahia, maria);\ 1) \\ (\sim prefProd(juan, sony\_hph);\ 1) \\ (spend(maria, 800);\ 1) \\ (price(600, sony\_hph, ncbahia);\ 1) \\ (stock(ncbahia, sony\_hph);\ 1) \\ (newProduct(sony\_hph);\ 1) \end{array} \right\}$$

The set Ψ contains information related to the users and the websites visited by the users: the literals $client(juan, ncbahia)$ and $trust(maria, juan)$ express that $juan$ is a client of the website *ncbahia* and $maria$ trusts $juan$, respectively. Moreover, the literal $\sim prefProd(juan, sony\_hph)$ conveys that *sony headphones* is not a preferred product by $juan$, and $stock(ncbahia, sony\_hph)$ indicates the fact that $ncbahia$ has *sony headphones* in stock and their price is \$600 ($price(600, sony\_hph, ncbahia)$). Finally, the literal $goodPrice(sony\_hph, ncbahia, maria)$ represents the fact that the price of *headphones sony* in *ncbahia* is at an acceptable price for *maria*.

The following set of defeasible rules with some initial weights can be used to model the agent's knowledge:

$$\Delta = \left\{ \begin{array}{l} (r1)(site(S, U) \leftarrow client(C, S), trust(U, C);\ 0.1) \\ (r2)(product(P, S, U) \leftarrow goodPrice(P, S, U);\ 0.4) \\ (r3)(\sim product(P, S, U) \leftarrow goodPrice(P, S, U), \\ \qquad trust(U, C), \sim prefProd(C, P); 0.3) \\ (r4)(av(P, S, U) \leftarrow selProd(P, U), spend(U, V), \\ \qquad price(C, P, S), V \geq C; 0.1) \\ (r5)(\sim av(P, S, U) \leftarrow selProd(P, U, S), \\ \qquad spend(U, V), price(C, P, S), V < C; 0.2) \end{array} \right\}$$

The first defeasible rule represents one tentative reason for recommending a sale website: "if C is a client of S and the client U trusts C, there is a reason to recommend S to U". The second and third rules are used for establishing whether the product P offered by S is suggestible for U. The third rule states that: "if P has a good price, but it is not a preferred product by a trusted client, then P is not suggestible". Finally, the fourth and fifth rules are used to indicate whether the product P offered by S is available for the user U. The fourth rule is read: "if P is a product particularly selected for U and the cost of P is less than what U hopes to spend, then P is available for U".

In [21], a mechanism where the use of guards offers a particular way of associating conditions to the selection of a preference criterion was introduced. A guard is a set of literals γ, and a guard is satisfied in a state Ψ when for each literal $L \in \gamma$, we have $(L,\ \omega) \in \Psi$ for some weight ω. Here, we use guards to guide the choice of a priority order associated with a given context that depends on the world's current state instead of selecting a criterion as was proposed in [21].

We rely on defeasible argumentation for reasoning over the preconditions of actions. Every action is associated with a conditional-contextual expression E. Then the planner will use a particular preference order on defeasible rules obtained after evaluating the expression E. In fact, in its simplest form E can be just a priority context $\mathsf{prOrder}_E$, and in that case the priority order corresponding to this priority context is used to evaluate the preconditions. In general, E can be of the form $E = [\gamma : E_1; E_2]$, where γ is a guard and where E_1 and E_2 can be in turn either priority contexts or further conditional expressions. In such a case, E is evaluated as follows: if γ is satisfied in the state of the world, then E_1 is evaluated, otherwise, E_2 is evaluated. This evaluation procedure is recursively applied until a priority context is found. In the application example, we could consider two expressions:

- $E_1 = [\{newProduct(P)\} : \mathsf{trust}; \mathsf{price}]$ where

$$\mathsf{prOrder}_{\mathsf{trust}} = \left\{ \begin{array}{l} (r2; 0.6) \\ (r3; 0.9) \\ [\ldots] \end{array} \right\} \quad \mathsf{prOrder}_{\mathsf{price}} = \left\{ \begin{array}{l} (r2; 0.8) \\ (r3; 0.5) \\ [\ldots] \end{array} \right\}$$

- $E_2 = \mathsf{price}$

The first expression E_1 is interpreted in the following way: "if P is a new product on the market, then the trust-based priority context is applied; otherwise, the price-based priority context is applied". The second expression E_2 expresses that the price priority context should be applied.

Note that conditional expressions are evaluated only on the basis of the strict part of the knowledge base. In [21],

it is argued that restricting the guards to use only strict derivations has been a design choice. The rationale for using strict derivations for determining whether a guard is satisfied, is that there is no need for an auxiliary criterion to decide between conflicting information since the strict part is non-contradictory. The study of other alternatives and, given its complexities, falls out of the scope of this work.

Apart from its KB, and having defined the preference expressions, the agent will have a set of actions that it may use to change its world. We define an action by a tuple $\langle \mathsf{A}, \mathsf{X}, \mathsf{P}, E \rangle$, where A is the name of the action, $\mathsf{X} = \{X_1, X_2, \ldots, X_n\}$ is a set of literals representing consequences of executing A, $\mathsf{P} = \{P_1, P_2, \ldots, P_n\}$ is a set of literals representing preconditions for A, and E is a conditional-preference expression under which to execute A. For instance, we can have the following actions:

- *offerProd(P,S,U)*: offer a product P to U that is sold from website S. The product P must be available for U.
- *chSite(S,U)*: choose a shopping website S for the user U. The website S must be recommended according to the user's preferences U.
- *chProd(P,U)*: choose a product P for the user U. The product P must be suggested according to the user's preferences.

Checking whether an action $\langle \mathsf{A}, \mathsf{X}, \mathsf{P}, E \rangle$ can be executed involves checking its applicability *i.e.,* checking whether the literals of the set P can be warranted by the updated program (Ψ', Δ'), resulting from evaluating previous actions, and of course the warrant of these literals will depend on the current priority context selected. Since a priority selection mechanism based on the current world state is used, a particular action may be associated with different contextual priorities in separate circumstances.

Given that the execution of an action results in a new state, other actions could be applicable over this state. That situation generates an *applicable sequence of actions*; when a goal state is reached through this sequence, it is called a *plan* for that goal.

Once the planning domain and preference expressions have been defined, the next step involves specifying the planning problem. We define a preference-based planning problem as the tuple $(\Psi, \Delta, \mathbf{A}, \mathbb{C}, \mathsf{M})$, where $\mathbb{C}$ is a set of priority contexts, and M is a set of literals representing an agent's goals. The agent satisfies its goals when, through the execution of a sequence of actions, it reaches a state Ψ' where each literal of M is warranted by the corresponding updated program (Ψ', Δ').

In what follows, Fig. 2 shows how the plan for the goal $sCart\_in(sony\_hph, maria)$ given by the sequence of actions $\mathsf{S} = [chSite(ncbahia, maria), chProd(sony\_hph, maria), offerProd(sony\_hph, ncbahia, maria)]$ is generated by an extension of the algorithm APOP, a Partial Order Planning algorithm based on Defeasible Logic Programming (DeLP) introduced in [3], that considers the conditional preference expressions proposed. The literal $sCart\_in(sony\_hph, maria)$ expresses that $sony\_hph$ is a product placed in *maria*'s virtual shopping cart. Consider a planner agent with the following actions:

Action $\langle \mathsf{A}_1, \mathsf{X}_1, \mathsf{P}_1, E_2 \rangle$ where
$\mathsf{A}_1 = chSite(S, U)$
$\mathsf{X}_1 = \{selSite(S, U)\}$
$\mathsf{P}_1 = \{site(S, U)\}$
$E_2 = \mathsf{price}$

Action $\langle \mathsf{A}_2, \mathsf{X}_2, \mathsf{P}_2, E_2 \rangle$ where
$\mathsf{A}_2 = chProd(P, U)$
$\mathsf{X}_2 = \{selProd(P, S, U)\}$
$\mathsf{P}_2 = \{selSite(S, U),$
$stock(S, P),$
$product(P, S, U)\}$
$E_2 = \mathsf{price}$

Action $\langle \mathsf{A}_3, \mathsf{X}_3, \mathsf{P}_3, E_1 \rangle$ where
$\mathsf{A}_3 = offerProd(P, S, M)$
$\mathsf{X}_3 = \{sCart\_in(P, U)\}$
$\mathsf{P}_3 = \{av(P, S, U)\}$
$E_1 = [\{newProduct(P)\} : \mathsf{trust}; \mathsf{price}]$

Next, our example scenario will serve for illustrative purposes to show how APOP can be extended to handle the conditional preference expressions presented before. As in the algorithm presented in [3], the extended partial order planning proposed here starts with an initial partial plan containing two steps: a *start* step whose effects encode an initial state, and a *finish* step whose preconditions encode the agent's goals to be achieved. The algorithm completes this initial plan with new steps (actions or arguments) until all steps's preconditions are warranted. The conditional-preference expression used in the initial step is assumed to be an empty expression. On the other hand, the *finish* step includes the conditional expression used to warrant the literals at the end of the epistemic plan.

In Figure 2, the labeled square nodes are used for action step specifications. The literals that appear below an action step represent the action step's preconditions, and the literals that appear above represent its effects. The literal that appears on the right side of an action step represents a priority context obtained from the conditional-preference expression associated with the action. Argument steps are depicted by triangles labeled with the argument name, and a literal representing the argument's conclusion in the top of the triangle. On the other hand, the solid arrows represent *causal links* of the plan used to link an action step effect with a precondition of another action step, or with a literal in the base of an argument step. The solid arrows that link the conclusion of an argument step and a precondition of an action step represent the plan's support links. Note that *ordering constraints* are represented by dashed arrows, and they allow establishing an order between steps. Arguments will use a graphical representation without their weights associated to simplify the notation in figures.

In Figure 2-(a), the *finish* action step has one unsatisfied precondition $sCart\_in(sony\_hph, maria)$. Note that the action *offerProd* is the only available action that can be used to satisfy the precondition. Thus, *offerProd* is added (Fig. 2-(b)) to the plan by the planning process and its precondition becomes a subgoal to be achieved under the priority context trust. Observe that none of the available

actions achieve $av(sony\_hph, ncbahia, maria)$. Nevertheless, from the rules Δ, it is possible to construct the undefeated argument $\mathcal{A}_1$ that supports $av(sony\_hph, ncbahia, maria)$. Then, $\mathcal{A}_1$ is selected, and the set of literals appearing in the body of rules conforming $\mathcal{A}_1$ become new subgoals (Fig. 2-(c)). In particular, the literals $spend(maria, 800)$ and $price(600, sony\_hph, ncbahia)$ are satisfied by the *start* step, whereas $selProd(sony\_hph, maria, ncbahia)$ is the effect of the action $chProd(sony\_hph, maria)$. Thus, a new step $chProd$ is added and now the preconditions $selSite(ncbahia, maria)$, $stock(ncbahia, sony\_hph)$, and $product(sony\_hph, ncbahia, maria)$ must be satisfied under preferences defined in the context price. The literal $stock(ncbahia, sony\_hph)$ is satisfied by the *start* step and from Δ it is possible to construct the argument $\mathcal{A}_2$, that is undefeated, supporting the subgoal $product(sony\_hph, ncbahia, maria)$. The literal $goodPrice(sony\_hph, ncbahia, maria)$ in the base of defeasible rule of $\mathcal{A}_2$ is achieved by the *start* step, whereas $selSite(ncbahia, maria)$ is the consequence of action $chSite(ncbahia, maria)$. Therefore, the corresponding step $chSite$ is added to the plan. Observe that steps $chSite$ and $chProd$ are configured to use the same priority context. From Δ it is posible to construct the undefeated argument $\mathcal{A}_3$ for the precondition $site(ncbahia, maria)$. Finally, the literal $site(ncbahia, maria)$ in the body of rule conforming $\mathcal{A}_3$ is satisfied by the *start* step, and the plan $\mathsf{S} = [\mathsf{A}_1, \mathsf{A}_2, \mathsf{A}_3]$ is formulated, and the product *headphones sony* is placed in *maria*'s virtual shopping cart.

To provide explanations for plans or the choice of a particular action, we propose explanations based on the approach followed by [22], where natural language explanations are obtained based on the structure of the arguments involved, replacing the logical structure to its corresponding colloquial interpretation. For instance, given the argument $\mathcal{A}_3$

$$\mathcal{A}_3 = \left\{ \begin{array}{c} site(ncbahia, maria) \leftarrow client(juan, ncbahia), \\ trust(maria, juan); 0.1) \end{array} \right\}$$

supporting the precondition $site(ncbahia, maria)$, such explanation could be:

Explanation: "the website $ncbahia$ is recommended to $maria$ because $juan$ is client of $ncbahia$ and $maria$ trusts $juan$".

Such explanations can then be offered to users upon request [10]. For space reasons, and since the user interface does not differ from typical ones in similar systems, we do not include details of this stage here.

IV. Conclusions and Future Work

The main contribution of this work has been to introduce an architecture for defeasible argumentation-based epistemic planning and the development of a methodological guideline for the analysis and design of planners capable of: (i) constructing plans from an incomplete or inconsistent knowledge base making combined use of argumentation and actions to perform this task; (ii) providing the possibility of adapting the preferences upon which preconditions of actions should be evaluated; (iii) expressing context-dependent preferences, *i.e.,* preferences over defeasible knowledge specified in a particular context; and (iv) providing the first step in proceeding towards an extension of the APOP algorithm to consider conditional expressions. The state of the art of DAEPs focuses on using defeasible reasoning over complete plans and for determining which actions are applicable in a given state, without including mechanisms to handle preferences in the process. Therefore, our work is an initial approach to investigating software development tailored to this type of systems. Future work involves further evolving these aspects towards a formal and robust software development methodology for creating high-quality DAEPs with mechanisms to express preferences. There are several research lines that we also plan to follow. When conditional expressions are used, an action consequences might interfere with the guards appearing in these expressions. A promising issue is as future work to analyze different type of interferences arising when conditional expressions are used. Finally, one of our future goals is to broaden the presented framework considering alternatives to model action consequences, possibly using defeasible knowledge.

Acknowledgments

The authors are indebted to the anonymous referees for their helpful insights and to Guillermo R. Simari for inspiring discussions. Teze acknowledges partial support from Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Universidad Nacional del Sur (UNS), Institute for Computer Science and Engineering (UNS-CONICET), and Universidad Nacional de Entre Ríos (UNER), Argentina. Godo acknowledges the AppPhil project (funded by CaixaBank, RecerCaixa 2017) and the Spanish projects RASO (TIN2015-71799-C2-1-P) and ISINC (PID2019-111544GB-C21).

References

[1] T. Bolander, "A gentle introduction to epistemic planning: The DEL approach," in *Proceedings of the Ninth Workshop on Methods for Modalities, M4M@ICLA 2017, Indian Institute of Technology, Kanpur, India, 8th to 10th January 2017*, ser. EPTCS, S. Ghosh and R. Ramanujam, Eds., vol. 243, 2017, pp. 1–22.

[2] I. Rahwan and G. R. Simari, *Argumentation in Artificial Intelligence*, 1st ed. Springer Publishing Company, Incorporated, 2009.

[3] D. R. García, A. J. García, and G. R. Simari, "Defeasible reasoning and partial order planning," in *Foundations of Information and Knowledge Systems, 5th International Symposium, FoIKS 2008, Pisa, Italy, February 11-15, 2008, Proceedings*, 2008, pp. 311–328.

[4] L. Amgoud, S. Parsons, and L. Perrussel, "An argumentation framework based on contextual preferences." in *Proceedings of the 3rd International Conference on Formal and Applied Practical Reasoning, FAPR '00, 2000*, 2000, pp. 59–67.

[5] J. L. Pollock, "The logical foundations of goal-regression planning in autonomous agents," *Artificial Intelligence*, vol. 106, no. 2, pp. 267–334, 1998.

[6] J. C. L. Teze, A. Perelló-Moragues, L. Godo, and P. Noriega, "Practical reasoning using values: an argumentative approach based on a hierarchy of values," *Annals of Mathematics and Artificial Intelligence*, vol. 87, no. 3, pp. 293–319, aug 2019.

[7] S. Pajares Ferrando and E. Onaindia, "Defeasible argumentation for multi-agent planning in ambient intelligence applications," in *International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2012, Valencia, Spain, June 4-8, 2012 (3 Volumes)*, W. van der Hoek, L. Padgham, V. Conitzer, and M. Winikoff, Eds. IFAAMAS, 2012, pp. 509–516.

[8] S. Pajares-Ferrando and E. Onaindia, "Defeasible-argumentation-based multi-agent planning," *Inf. Sci.*, vol. 411, pp. 1–22, 2017.

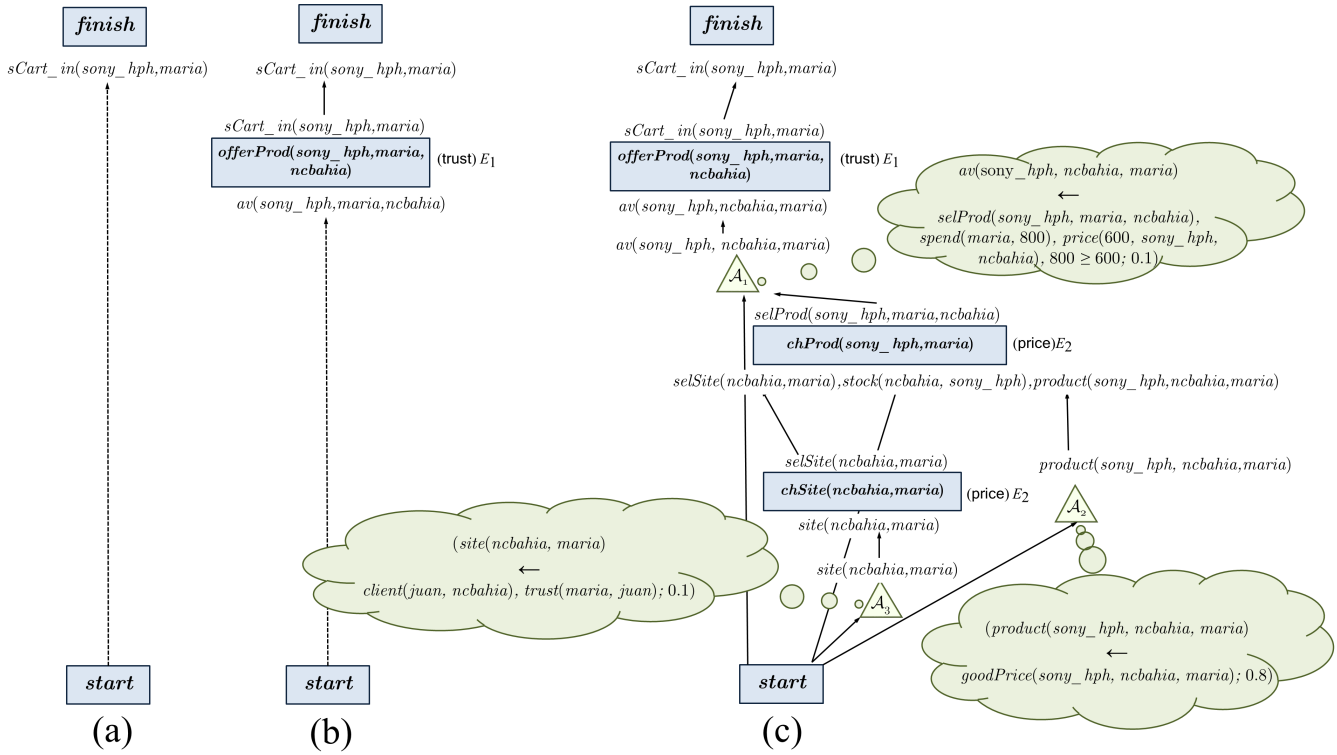

Fig. 2: Different partial plans for the application example.

[9] P. Pardo and L. Godo, "A temporal argumentation approach to cooperative planning using dialogues," *J. Log. Comput.*, vol. 28, no. 3, pp. 551–580, 2018.

[10] N. Oren, K. van Deemter, and W. W. Vasconcelos, "Argument-based plan explanation," in *Knowledge Engineering Tools and Techniques for AI Planning*, M. Vallati and D. E. Kitchin, Eds. Springer, 2020, pp. 173–188.

[11] S. Kaci, *Working with Preferences: Less Is More*, ser. Cognitive Technologies. Springer, 2011.

[12] A. J. García and G. R. Simari, "Defeasible logic programming: An argumentative approach," *Theory and Practice of Logic Programming (TPLP)*, vol. 4, no. 1-2, pp. 95–138, 2004.

[13] J. A. Baier and S. A. McIlraith, "Planning with preferences," *AI Magazine*, vol. 29, no. 4, pp. 25–36, 2008.

[14] Z. Shams, M. D. Vos, N. Oren, and J. Padget, "Argumentation-based reasoning about plans, maintenance goals, and norms," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 14, no. 3, pp. 1–39, mar 2020.

[15] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artif. Intell.*, vol. 77, no. 2, pp. 321–358, 1995.

[16] J. Pearl, "Graphical models for probabilistic and causal reasoning," in *Computing Handbook, Third Edition: Computer Science and Software Engineering*, T. F. Gonzalez, J. Diaz-Herrera, and A. Tucker, Eds. CRC Press, 2014, pp. 44: 1–24.

[17] A. E. Gerevini and A. Saetti, "An interactive tool for plan generation, inspection, and visualization," in *Knowledge Engineering Tools and Techniques for AI Planning*. Springer, 2020, pp. 127–155.

[18] T. Alsinet, C. I. Chesñevar, L. Godo, and G. R. Simari, "A logic programming framework for possibilistic argumentation: Formalization and logical properties," *Fuzzy Sets and Systems*, vol. 159, no. 10, pp. 1208–1228, 2008.

[19] V. Lifschitz, "Foundations of logic programs," in *Principles of Knowledge Representation*, G. Brewka, Ed. USA: Center for the Study of Language and Information, 1997, pp. 69–128.

[20] T. Alsinet, R. Béjar, L. Godo, and F. Guitart, "RP-DeLP: a weighted defeasible argumentation framework based on a recursive semantics," *Journal of Logic and Computation*, vol. 26, no. 4, pp. 1315–1360, feb 2014.

[21] J. C. Teze, S. Gottifredi, A. J. García, and G. R. Simari, "An approach to generalizing the handling of preferences in argumentation-based decision-making systems," *Knowledge-Based Systems*, p. 105112, 2019.

[22] C. E. Briguez, M. C. Budán, C. A. D. Deagustini, A. G. Maguitman, M. Capobianco, and G. R. Simari, "Argument-based mixed recommenders and their application to movie suggestion," *Expert Systems with Applications*, vol. 41, no. 14, pp. 6467–6482, 2014.

Juan Carlos Teze is a professor at Universidad Nacional de Entre Ríos, and a researcher at the Institute of Computer Science and Engineering (UNS – CONICET), Argentina. His main interests are in Artificial Intelligence, with a particular focus on representation and reasoning with preferences. He obtained his Ph.D. from Universidad Nacional del Sur (Bahía Blanca, Argentina), in the area of Artificial Intelligence. E-mail: carlos.teze@uner.edu.ar.

Lluis Godo is a Research Professor at the Artificial Intelligence Research Institute (IIIA) of the Spanish National Research Council (CSIC), Barcelona, Spain. He obtained his PhD in Mathematics from the Technical University of Catalunya (1990). His main research interests include logics for Artificial Intelligence, in particular graded uncertainty reasoning formalisms, mathematical fuzzy logic, and argumentation systems. He is an EurAI Fellow and an IFSA Fellow. E-mail: godo@iiia.csic.es