Defeasible Argumentation for Multi-Agent Planning in Ambient Intelligence Applications

Sergio Pajares Ferrando
Dpto. de Sistemas Informáticos y Computación
Universitat Politècnica de València
Camino de Vera, s/n, 46022 Valencia, Spain
spajares@dsic.upv.es

Eva Onaindia
Dpto. de Sistemas Informáticos y Computación
Universitat Politècnica de València
Camino de Vera, s/n, 46022 Valencia, Spain
onaindia@dsic.upv.es

ABSTRACT

This contribution presents a practical extension of a theoretical model for multi-agent planning based upon DeLP, an argumentation-based defeasible logic. Our framework, named DeLP-MAPOP, is implemented on a platform for open multi-agent systems and has been experimentally tested, among others, in applications of ambient intelligence in the field of health-care. DeLP-MAPOP is based on a multi-agent partial order planning paradigm in which agents have diverse abilities, use an argumentation-based defeasible reasoning to support their own beliefs and refute the beliefs of the others according to their knowledge during the plan search process. The requirements of Ambient Intelligence (AmI) environments featured by the imperfect nature of the context information and heterogeneity of the involved agents make defeasible argumentation be an ideal approach to resolve potential conflicts caused by the contradictory information coming from the ambient agents. Moreover, the ability of AmI systems to build a course of action to achieve the user's needs is also a claiming capability in such systems. DeLP-MAPOP shows to be an adequate approach to tackle AmI problems as it gathers together in a single framework the ability of planning while it allows agents to put forward arguments that support or argue upon the accuracy, unambiguity and reliability of the context-aware information.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems

General Terms

Algorithms, Experimentation

Keywords

Defeasible Argumentation, Multi-Agent Planning, Ambient Intelligence.

1. INTRODUCTION

Ambient Intelligence (AmI) integrates concepts ranging from Ubiquitous Computing to Artificial Intelligence with the vision that technology will become invisible, embedded in our natural surroundings, present whenever we need it, and adaptive to users [1].

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), June, 4–8, 2012, Valencia, Spain.

In AmI environments, people are surrounded with networks of embedded intelligent devices that can sense the available context information, anticipate, and perhaps adapt to their needs. In this contribution, we handle these requirements by modeling ambient agents as entities which manage a portion of the AmI environment, i.e. they are responsible for one or more devices. Due to the imperfect nature of the context and the heterogeneity of ambient agents, whose different viewpoints lead them to infer different assumptions about the user's current situation, ambient agents, as distributed autonomous software entities, are required to engage in interactions, argue with one another, and make agreements, individually or collectively, while responding to changing circumstances of the ambient environment. For this reason, ambient agents are being advocated as a next-generation model for engineering complex distributed systems such as AmI systems. The aim in AmI is to make the interaction between users and the smart environment easy.

Defeasible is the opposite of irrefutable or indisputable. A defeasible piece of information is a non-demonstrative piece of information that is acknowledged to be able to fail or be corrected. Defeasible reasoning is usually realized as a rule-based approach for reasoning with incomplete and inconsistent information through the use of rules that may be defeated by other rules. Defeasible reasoning has been successfully used in AmI applications [2]. On the other hand, **Defeasible Argumentation**, which has recently become a very active research field in computer science [3], is a form of defeasible reasoning that emphasizes the notion of argument. An argument is a chain of reasoning that concludes one piece of information (conclusion) on the basis of some other pieces of information (premises). Thus, defeasible argumentation can be viewed as a powerful tool for reasoning about inconsistent information through a rational interaction of arguments for and against some conclusion derived by an ambient agent. Defeasible argumentation has also been successfully proved in AmI applications [4].

The defeasible logic programming formalism DeLP [5] is one of the most popular approaches to build defeasible argumentation. Our framework, DeLP-MAPOP, builds upon DeLP to implement the defeasible argumentation mechanism. The key element of DeLP are defeasible rules (Head $\prec$ Body), which are used to represent a deductive relation between pieces of knowledge that could be defeated once another piece of knowledge is considered. For instance, a defeasible rule like *emergency* $\prec$ *patient-fever* denotes that an ambient agent believes that if the monitoring system returns the patient has fever then there are provable reasons to declare an emergency. The defeasible rule *∼emergency* $\prec$ *{normal-pulse, conscious, correct-breathing}* provides reasons to believe the contrary, in whose case we say that the first piece of information is acknowledged to fail in case *{normal-pulse, conscious, correct-breathing}* hold in the context. However, assuming that another ambient agent knows that the patient is vomiting blood, i.e. {*bloody-vomit*} holds in the context, then it might derive the patient has not a normal pulse by following the defeasible rules {$\sim$*normal-pulse* $\prec$ *internal-bleeding*; *internal-bleeding* $\prec$ *bloody-vomit*}, which represents an attack to the defeasible rule whose conclusion is $\sim$*emergency*. Thus, arguments (combinations of defeasible rules and facts) for conflicting pieces of information are built, and then compared to decide which one prevails.

Planning is a desired ability in AmI systems to achieve a goal-oriented behavior, i.e. to decide the course of action to meet the needs of the specific application, for instance, stabilizing a patient in a home-care system. Planning has been used in some AmI applications for monitoring and responding to the needs of a diabetic patient [6]. Particularly, the work in [6] presents a centralized planner that manages distributed capabilities as it assumes that some agents do not have planning capabilities. In this case, an agent is implemented as a device, which prevents the agent from taking responsibilities in building the plan due to its limitations in processing and communication; for example, a cell phone could not be able to autonomously plan to call a doctor given that other devices detected that a user in the environment is ill [6]. However, in our contribution an ambient agent is executed on an independent host and can encompass several devices. This increases the communication capacity as well as autonomy and endow agents with the necessary abilities to pose a goal and build a plan for this goal. This approach allows us to address many real applications where the capabilities to perceive the context and perform the actions are distributed across agents. **Multi-Agent Planning** (MAP) applied to an AmI environment is intended as the ability of a team of ambient agents to build collaboratively a plan of actions that, when performed in the AmI context, meets the needs and goals of the application.

Partial Order Planning (POP) is a suitable planning approach to address the requirements derived from a distributed planning thanks to the application of a non-sequential behaviour and the least commitment principle [7]. This is evidenced by the fact that most existing architectures for integrating planning with execution, information gathering and scheduling are based on partial order planners. In [8], authors argue that POP-based frameworks offer a more promising approach for handling domains with durative actions and temporal and resource constraints as compared to other planning approaches. In fact, most of the known implementations of planning systems capable of handling temporal and durative constraints (e.g. NASA's RAX [9]) are based on the POP paradigm. Even for simple planning tasks, partial order planners offer a higher degree of execution flexibility. For these reasons, this work is based on Multi-Agent Partial Order Planning (MAPOP).

A extension of POP with DeLP-style argumentation, denoted as DeLP-POP, was introduced in [10], where both actions and arguments may be used to enforce some goal, if their conditions (are known to) apply and arguments are not defeated by other arguments applying. Unlike actions, arguments are not only introduced to intentionally support some step of a plan, but they are also presented to defeat or defend other supporting arguments in the plan. When actions and arguments are combined in a partial order plan, new types of interferences or threats appear [10] which need to be identified and resolved to obtain valid plans. Finally, the work in [11, 12, 13] proposes an extension of the DeLP-POP to a multi-agent environment. Specifically, it proposes a dialogue for argumentative plan search, by which agents exchange plan proposals and arguments for or against such proposals. To the best of our knowledge, these theoretical works have neither been implemented nor tested on real-world domains such as AmI applications.

This contribution presents DeLP-MAPOP, a system that combines, implements and tests features like multi-agent defeasible argumentation and multi-agent planning in AmI applications. DeLP-MAPOP develops and implements an extended and refined version of the framework presented in [12]; DeLP-MAPOP is applied and experimentally tested in an AmI environment, it extends the agents' knowledge bases and the dialogues during the plan search and it offers a new classification of planning interferences. The remainder of this paper is divided as follows. First, we introduce the basic elements of the system; then we present the MAP protocol applied in an AmI scenario to deal with a person suffering from a heart disease. Next, the experiments carried out to validate the present work are described and analyzed. Finally, we conclude and present some directions for future work.

2. COMPONENTS OF THE SYSTEM

In this section, we provide definitions for the notions of ambient agent, context information, planning task, argument versus action and plan, that will be later used for the definition of the DeLP-MAPOP protocol.

2.1 Ambient Agents

In DeLP-MAPOP, ambient agents act as planning agents with different beliefs, capabilities and preferences. Thus, we assume the capabilities to perceive the context, perform actions and derive new conclusions are distributed across ambient agents. Agents are managed and supervised by the Agent Management System (AMS) that is responsible for the following tasks: i) Exercising supervisory control over access to the multi-agent platform; it is responsible for authentication of resident ambient agents and control of registrations. ii) Discovering new user's needs generated directly by the user or indirectly by a smart device, which provides the input to a DeLP-MAPOP process in terms of goals to be reached. iii) When ii) occurs, the AMS agent gathers the ambient agents who will participate in the planning process and will return the action plan to satisfy the user's needs. For instance, a device that monitors the patient's heart's rate may detect the presence of arrhythmias by means of an electrocardiogram, a symptom that might entail a heart attack. In this case, the monitoring system generates the goal *patient-to-be-treated*, and communicates it to the AMS agent.

The knowledge of an ambient agent mainly comprises context information encoded as defeasible rules and initial facts, and context capabilities represented as planning actions.

2.2 Context information

The representation scheme used by DeLP-MAPOP to model components of the AmI environment is based on a state-variable representation, where variables map to a finite domain of values which represent the problem objects. A state-variable representation is equivalent to a classical planning representation in expressive power and it is also useful in non-classical planning problems as a way to handle numbers, functions and time. In this paper, we will restrict our attention to only non-numeric variables. Since actions change the state of the world and defeasible rules make assumptions about the state of the world, actions and defeasible rules are most naturally modeled as elements that change the values of the state variables. The variable-value pair $\langle v_i, vl_i \rangle$ denotes the value vl_i is assigned to the variable v_i. For instance, the variable-value pair $\langle$ *at-amb, pH* $\rangle$ indicates that the ambulance amb is located at the patient's home pH, that is, the value of the variable denoting the position of the ambulance is the patient's home.

In what follows, we define the set of elements used to represent the agent's context information. (i) the set of objects O that model the elements of the planning domain over which the actions and defeasible rules can act. (ii) the set of state variables V that are used to model the states of the world: each state variable $v_i \in V$ is mapped to a finite domain of mutually exclusive values D_{v_i}, where $\forall v_i \in V, D_{v_i} \subseteq O$. (iii) the initial state of the problem Ψ, which is a consistent set of variable-value pairs; a variable with no assigned value in the initial state is assumed to have an *unknown* value. (iv) the set of defeasible rules Δ, where each rule δ follows the form $\langle \mathsf{head}(\delta) \prec \mathsf{body}(\delta) \rangle$; if the set of variable-value pairs in $\mathsf{body}(\delta)$ is warranted, i.e. if variables have the specified values in the pair, then δ is applicable and for each $\langle v_i, vl_i \rangle$ that appears in the head of the rule, v_i is assigned the value vl_i. (v) A is the set of planning actions $\alpha = \langle \mathsf{P}(\alpha), \mathsf{X}(\alpha) \rangle$, where $\mathsf{P}(\alpha)$ is a set of preconditions encoded as variable-value pairs that must be satisfied in order to apply the effects in $\mathsf{X}(\alpha)$, also encoded as $\langle v_i, vl_i \rangle$.

2.3 Planning task

Each ambient agent $x \in \{\mathsf{Ag}_1 \ldots \mathsf{Ag}_n\}$ is initially endowed with **a planning task** $\mathbb{M}_x = (O_x, V_x, \Psi_x, \Delta_x, A_x, F_x, G)$ where:

1. O_x is the set of objects known by the agent x.
2. V_x is the set of variables managed by agent x to represent the agent's knowledge about the state of the world.
3. $\Psi_x = \{\langle v_i, vl_i \rangle \mid v_i \in V_x; vl_i \in D_{v_i}\}$ represents the partial view of the initial world state of agent x, i.e. the information that agent x knows about the initial state. We assume $\bigcup_{x \in \{\mathsf{Ag}_1 \ldots \mathsf{Ag}_n\}} \Psi_x$ is a consistent set.
4. Δ_x is a set of defeasible rules known by the agent x.
5. A_x is a set of planning actions known by the agent x.
6. F_x represents a consistent set of the agent-specific preferences $F_x \subseteq \{(a, d) \mid (a \in A_x), d \in [0, 100]\}$, where action a is preferred with the estimated interest degree d.
7. G is the set of global goals that represent the needs of a user in an AmI environment. G is expressed as a set of pairs variable-value thus indicating the value each variable is expected to assume in the final state. Unlike the rest of elements, G is known by all of the ambient agents.

2.4 Arguments versus Actions

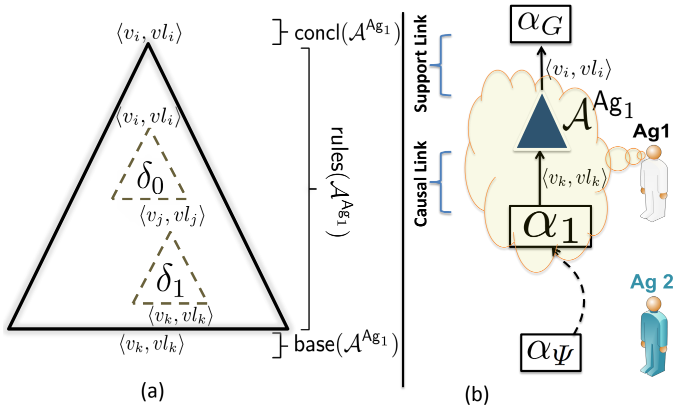
As we saw in the Introduction section, and based on the framework presented in [10], both actions and arguments may be used to enforce some task goal in DeLP-MAPOP. As illustrated in Figure 1 (a), an **argument** $\mathcal{A}$ for $\langle v_i, vl_i \rangle$ proposed by an ambient agent Ag_1, is denoted as $\mathcal{A}^{\mathsf{Ag}_1} = (\{\mathsf{concl}(\mathcal{A}^{\mathsf{Ag}_1})\}, \{\mathsf{rules}(\mathcal{A}^{\mathsf{Ag}_1})\})$, where $\mathsf{concl}(\mathcal{A}^{\mathsf{Ag}_1}) = \langle v_i, vl_i \rangle$ is the argument conclusion and $\mathsf{rules}(\mathcal{A}^{\mathsf{Ag}_1})$ is a subset of defeasible rules such that $\mathsf{rules}(\mathcal{A}^{\mathsf{Ag}_1}) \subseteq \Delta_x$. $\mathcal{A}^{\mathsf{Ag}_1}$ is consistent if there exists a defeasible derivation for $\langle v_i, vl_i \rangle$ from $\mathsf{base}(\mathcal{A}^{\mathsf{Ag}_1}) \cup \mathsf{rules}(\mathcal{A}^{\mathsf{Ag}_1})$, where $\mathsf{base}(\mathcal{A}^{\mathsf{Ag}_1})$ is the argument base, the set of <variable,value> that must be warranted in the agent's context information. The existence of an argument $\mathcal{A}^{\mathsf{Ag}_1}$ does not suffice to warrant its conclusion $\langle v_i, vl_i \rangle$, this depends on the interactions among arguments as we will see in Section 3.3. We semantically distinguish between **supporting arguments** (also known as argument steps) as the arguments specifically used to support some goal of the plan, and **attacking arguments** (also known as defeaters) which are only introduced to attack some argument step previously introduced in the plan.

The difference between assigning a value to a variable by an argument or by an action is that in the case of a planning action the value is indisputable because it reflects a modification stated in the problem domain modelling; however, the confirmation of a value assigned to a variable by an argument depends on the interaction with other attacking arguments.

2.5 Plans

In POP, a partial order plan Π is a set of partially ordered actions (denoted by the relation $\prec$) which actually encodes multiple linear plans. More specifically, a **plan** Π is a tuple $\Pi = (A(\Pi), \mathcal{AR}(\Pi), G(\Pi), \mathcal{OC}(\Pi), \mathcal{CL}(\Pi), \mathcal{SL}(\Pi))$, where $A(\Pi)$ denotes the set of action steps, $\mathcal{AR}(\Pi)$ represents the set of argument steps, $G(\Pi)$ is the task's common goals (the user's needs), $\mathcal{OC}(\Pi)$ is a set of ordering constraints, and $\mathcal{CL}(\Pi)$ and $\mathcal{SL}(\Pi)$ represent the sets of causal and support links, respectively. In POP, Ψ and G are encoded as dummy actions $\{\alpha_\Psi \prec \alpha_G\}$ where α_Ψ is also refereed to as the initial step of the plan and α_G to as the final step of the plan, with $\mathsf{X}(\alpha_\Psi) = \Psi$, $\mathsf{P}(\alpha_G) = G$, and $\mathsf{P}(\alpha_\Psi) = \mathsf{X}(\alpha_G) = \emptyset$.

Let $\langle v_i, vl_i \rangle$ be an open goal in Figure 1(b), motivated by some action step $\alpha_G \in A(\Pi)$, i.e. $\langle v_i, vl_i \rangle \in \mathsf{P}(\alpha_G)$; let $\langle v_k, vl_k \rangle$ be another open goal, motivated by some argument step $\mathcal{A}^{\mathsf{Ag}_1}$, i.e. $\langle v_k, vl_k \rangle \in \mathsf{base}(\mathcal{A}^{\mathsf{Ag}_1})$. Then, the goal $\langle v_i, vl_i \rangle \in \mathsf{P}(\alpha_G)$ must be supported by an argument, argument $\mathcal{A}^{\mathsf{Ag}_1}$ in Figure 1(b), which introduces a **support link** $(\mathcal{A}^{\mathsf{Ag}_1}, \langle v_i, vl_i \rangle, \alpha_G) \in \mathcal{SL}(\Pi)$, where $\mathcal{SL}(\Pi) \subseteq \Delta \times G(\Pi) \times A$. In contrast, the goal $\langle v_k, vl_k \rangle$ must be supported by an action, α_1 in Figure 1(b), which introduces a **causal link** $(\alpha_1, \langle v_k, vl_k \rangle, \mathcal{A}^{\mathsf{Ag}_1}) \in \mathcal{CL}(\Pi)$, where $\mathcal{CL}(\Pi) \subseteq A \times G(\Pi) \times \Delta$. Triangles in Figure 1(b) represent argument steps (i.e. arguments that support preconditions of action steps), while rectangles represent action steps (i.e. actions that support the basis of an argument step). Therefore, in this approach, goals must always be initially derived by some argument step, and an argument base must be satisfied by another action step (including the initial step). This way, a typical causal link in POP is now replaced by a causal link and a support link. Note this representation allows us to implicitly address *the qualification problem* [14] as every precondition of a planning action is now supported by an argument step rather than directly by an action effect. This way, agents may attack the fulfillment of such precondition if they believe that there exist other non-explicit conditions that prevent the supporting action from having its intended effects. This new conception of mandatorily supporting preconditions through argument steps gives rise to a new and unique notion of threat. Under this new perspective, the concept of argument-argument threat in [10, 12] is now replaced by a broader notion of argument-argument threat that covers all the interferences that arise between the elements of a plan in which the qualification problem is addressed through the use of argument steps. Depending on where these argument-argument threats occur in the plan, we will distinguish between **threats** (Section 3.2) and **attacks** (Section 3.3).



$\langle v_i, vl_i \rangle$
$\mathsf{concl}(\mathcal{A}^{\mathsf{Ag}_1})$
$\langle v_i, vl_i \rangle$
δ_0
$\langle v_j, vl_j \rangle$
δ_1
$\langle v_k, vl_k \rangle$
$\mathsf{rules}(\mathcal{A}^{\mathsf{Ag}_1})$
$\langle v_k, vl_k \rangle$
$\mathsf{base}(\mathcal{A}^{\mathsf{Ag}_1})$
(a)
Support Link
Causal Link
α_G
$\langle v_i, vl_i \rangle$
$\mathcal{A}^{\mathsf{Ag}_1}$
Ag1
$\langle v_k, vl_k \rangle$
α_1
Ag 2
α_Ψ
(b)

Figure 1: (a) An argument $\mathcal{A}^{\mathsf{Ag}_1}$ for $\langle v_i, vl_i \rangle$ by using two defeasible rules: $\delta_0 = \{\langle v_i, vl_i \rangle\} \prec \{\langle v_j, vl_j \rangle\}$ and $\delta_1 = \{\langle v_j, vl_j \rangle\} \prec \{\langle v_k, vl_k \rangle\}$, such that $v_i \neq v_j$ and $v_j \neq v_k$ and $\{v_i, v_j, v_k\} \subseteq V_x$; (b) An example of a partial plan.

3. MULTI-AGENT PLANNING PROTOCOL

First, we outline the procedure followed by the DeLP-MAPOP protocol that interleaves a planning stage, an argumentation stage and a selection stage. Given a set of global goals, G, that address the requirements of an AmI application, agents build their own planning task $\mathbb{M}_x$ so they can differently contribute to the construction of the joint solution plan. The starting point of the MAP protocol is an empty initial plan Π_0 and the output is the solution plan. Once checked the plan Π is not a solution, the first step is to select an open goal $\Phi \in G(\Pi)$ of the planning task for resolution (choose[1] step in Algorithm 1). Then it comes the planning stage (PROPOSALS step in Algorithm 1) where agents put forward and exchange different partial order plans that would potentially solve Φ. Following, agents get involved in an argumentative dialogue (EVALUATION step in Algorithm 1) in which they expose their arguments for or against each of the proposals. This evaluation process performs a *warranty procedure* to determine which proposals do not receive attacks or, otherwise, the received attacks do not succeed. Subsequently, ambient agents reach an agreement as to which about the next partial plan and they continue the search exploration (SELECTION step in Algorithm 1). The process is repeated until a solution plan is found.

Algorithm 1: Multi-agent planning protocol overview.

```
input  : The initial plan $\Pi_0 := \{\alpha_\Psi \prec \alpha_G\}$.
output: The solution plan $\Pi$.

$\Pi := \Pi_0$
while $\Pi <> null$ do
    if $G(\Pi) = \emptyset$ then
        return $\Pi$ [It is a plan solution.]
    else
        choose $\Phi \in G(\Pi)$;
        $\mathrm{Ref}(\Pi, \Phi) := \mathrm{PROPOSALS}(\Pi, \Phi)$;
        [Each plan $\Pi_r$ of the set $\mathrm{Ref}(\Pi, \Phi)$ is a choice
        (partial-order plan) extending $\Pi$.]
        if $\mathrm{Ref}(\Pi, \Phi) = \emptyset$ then
            [Backtracking process.]
        else
            $\mathrm{EVALUATION}(\mathrm{Ref}(\Pi, \Phi))$;
            $\Pi := \mathrm{SELECTION}()$;
return fail; [Not exists plan.]
```

The state-variable representation used in DeLP-MAPOP is based on the latest PDDL (Planning Domain Definition Language) version, PDDL3.1 [15], which was introduced in the context of the 2008 International Planning Competition. Here, we extend the language PDDL3.1 for supporting the specification of defeasible rules and the ambient agent's preferences. Moreover, our language allow us to specify binary variables. A state variable v_i is interpreted in PDDL3.1 as a function that represents a characteristic shared by some of the objects that define the problem. v_i is a tuple that takes the following form $v_i = (vN_i\ p_1 \ldots p_n)$, where vN_i is the unique variable's name and $p_1 \ldots p_n$ are the objects as input parameters of the function. For instance, let *pos-t11* be a variable that indicates the current position of the medical team *t11*; this variable is encoded in PDDL3.1 through the function *(pos t11)*, where '*pos*' is the function name and *t11* is the function parameter. An assignment of a value vl_i to a variable v_i in PDDL3.1 is denoted by $(\mathtt{assign}\ \mathtt{v_i}\ \mathtt{vl_i})$; and the comparison operation is represented by $(=\ \mathtt{v_i}\ \mathtt{vl_i})$. We also allow to express multi-valued variables for ease of coding, denoted by $(\mathtt{member}\ \mathtt{v_i}\ \mathtt{vl_i})$. For simplicity, we will use the notation <variable,value> in the explanations and use the PDDL3.1 language only to show the encoding of the defeasible rules and planning actions of the planning task. All of these encodings will be shown in a framed box labeled with the caption name Listing.

[1]The open goal Φ is selected as the most costly open goal according to a reachability analysis of the variables.

3.1 Overview of the Application Scenario

This section provides a brief overview of the AmI application upon which the framework DeLP-MAPOP is applied. The purpose is to motivate the interest of this type of applications as well as the utilization of a defeasible planning model to carry out the necessary operations to fulfill the user's need at a specific time.

Nowadays, more and more patients are suffering heart diseases which is the main cause of premature death. The monitoring of people suffering heart failure is currently a challenge for AmI systems. The work in [6] presents a first approach to use an AmI system with centralized planning capabilities for assisting patients suffering diabetics problems. Here, we assume that the patient's home is equipped with appropriate technologies to create the AmI environment. The patient is monitored with a system, in the form of a bracelet, which collects the patient's physical activity and wirelessly transmits it to a device responsible for monitoring patient's heart rate. When a need is detected by this device, e.g. an extremely lower level of a patient's physical activity which may end up in a heart attack, the AmI environment executes DeLP-MAPOP for assisting the patient until the health services arrive to the patient's home.

In this application, we have the following ambient agents: a communication agent in charge of using telecommunication devices such as a cell telephone to call the emergency services; the assistant agent, who is responsible for controlling an automated external defibrillator, an activity tracking device, a position tracking device, etc. to interact with both the environment and the user; and the transport agent, whose main function is to guide the ambulance/helicopter to follow the best path to reach the patient's home. Agents have different capabilities according to their role so they contribute to the overall plan with different actions accordingly. However, we assume that agents' beliefs concern any aspect of the context information and so agents can make assumptions on the current status of the application regarding any type of information. That is, beliefs are not necessarily related to the planning capabilities of the agent, they can refer to any aspect of the AmI environment. The hospitals' preferences are associated with the transport-agent specific preferences, while the patient's preferences are related to the specific preferences of the assistant agent. For space reasons, we omit the specification of the planning task of each ambient agents.

3.2 Plan proposals process

At the PROPOSALS stage, agents generate their refinements $\mathrm{Ref}(\Pi, \Phi)$ to solve an open goal Φ in a partial plan Π, similarly to a plan-space planning process that builds a POP tree, except that each refinement or successor of Π may be now generated by a different agent. Another distinguishing characteristic of the partial order plans generated in DeLP-MAPOP is that they also contain argument steps, as explained in section 2.5, to support action preconditions; this argument structure formed in each partial plan will be later used in the EVALUATION process. The PROPOSALS stage finishes when all agents have made their plan proposals at their turn and these are communicated to the rest of agents. Then, agents update their set of actions with the information appearing in the refinements proposed by the other agents.

Let's suppose an ambient agent Ag_1 who has transport capabilities and knows there are three hospitals in the city $\{H1, H2, H3\}$. Each hospital disposes of two ambulances from *{a11, a12 . . ., a32}* (one equipped with an Advanced Life Support (ALS) equipment, and the other equipped with a Basic Life Support (BLS) equipment) and one emergency helicopter from *{h1, . . . h3}*. Moreover, Ag_1 knows there are always two emergency medical teams from the set *{t11, t12 . . ., t32}* on call in each hospital: one handles the ALS emergency equipment, and is formed by an ambulance driver, a nurse and a physician; the other handles the BLS equipment and is formed by an ambulance driver and a nursing assistant. Ag_1 also has the defeasible rule specified in Listing 1 and the planning action shown in Listing 2, among others. Note that the new location of the ambulance and the medical-team are generated through the defeasible rule moved-medical-assistance, which is embedded in an argument whose base must be supported by the effects of the action moving-medical-assistance. This allows agents to intervene during the argumentative dialogue in the EVALUATION stage to defeasibly attack the intended effects of the planning action; that is, in case agents have beliefs that make them conclude that the action would not achieve its expected effects.

```
(:def-rule moved-medical-assistance
    :parameters(?a - ambulance
    ?a1 address-hospital ?a2 - address-patient-home
    ?m - medical-team)
    :head (and (assign (at ?a) ?a2)
               (assign (pos ?m) ?a2))
    :body (and (= (moved-amb ?a ?a1) ?a2)
               (= (moved-team ?m ?a1) ?a2)))
```

Listing 1: The body of the defeasible rule matches the effects of the action moving-medical-assistance to deal with the qualification problem.

```
(:action moving-medical-assistance
   :parameters (?a - ambulance
    ?a1 address-hospital ?a2 - address-patient-home
    ?m - medical-team ?t - support-type)
    :effect (and (assign (moved-amb ?a ?a1) ?a2)
                 (assign (moved-team ?m ?a1) ?a2))
    :precondition (and (member (link ?a1) ?a2)
                       (member (type ?t) ?m)
                       (member (contains ?t) ?a)
                       (= (at ?a) ?a1)
                       (= (pos ?m) ?a1)))
```

Listing 2: An action for moving an ambulance from a location to other one.

Let pH be the patient's home. If Ag_1 is asked to solve the open goal $\mathsf{P}(\alpha_G) = \langle$ *at-?a, pH* $\rangle$ ('?a' is an ambulance) generated by the AMS agent to assist the patient, Ag_1 generates at least 6 refinement plans (3 hospitals * 2 ambulances) by using Listings 1 and 2 at the PROPOSALS stage. One of these proposed refinement plan generated is $\Pi_r^{\mathsf{Ag}_1}$, such that $\mathcal{OC}(\Pi_r^{\mathsf{Ag}_1}) = \{\alpha_\Psi \prec \alpha_1;\ \alpha_1 \prec \mathcal{A}^{\mathsf{Ag}_1}; \mathcal{A}^{\mathsf{Ag}_1} \prec \alpha_G\}$, as shown graphically in Figure 1(b); in this particular example:

- $\mathsf{concl}(\mathcal{A}^{\mathsf{Ag}_1}) = \{\langle$ *pos-t11, pH* $\rangle$, $\langle$ *at-a11, pH* $\rangle\}$ matches $\mathsf{P}(\alpha_G)$.
- $\mathsf{X}(\alpha_1) = \{\langle$ *moved-amb-a11-H1, pH* $\rangle$, $\langle$ *moved-team-t11-H1, pH* $\rangle\}$ matches $\mathsf{base}(\mathcal{A}^{\mathsf{Ag}_1})$.

Therefore, argument $\mathcal{A}^{\mathsf{Ag}_1}$ is indirectly deriving the effects of the action α_1. However, unlike non-argumentative MAP systems, in DeLP-MAPOP the open goal $\langle$ *at-?a, pH* $\rangle$ can also be derived by means of an argument that an agent, say Ag_2, puts forward to indicate, that according to its knowledge, ambulance *a31* is already at the patient's home. The base of this argument may be supported with the information provided by an ambulance position tracking device which allows Ag_2 to infer that an ambulance is already located at the patient's home. As the rest of agents do not own this information, they would claim for the inclusion of an action that moves an ambulance to the indicated place. Therefore, unlike classical planning, the argumentation mechanism in DeLP-MAPOP enables supporting an open goal with the context information of an agent without having to necessarily include an action to satisfy such goal, which results in less costly plans. The next stage would show the procedure to guarantee that a goal is satisfactorily warranted by an argument.

3.3 Plan evaluation process

At the EVALUATION stage, agents become engaged in a number of argumentative dialogues aimed at evaluating the guarantee of a successful execution of a plan proposal, i.e the possibility that the actions' intended effects or the derived information, both represented as argument steps in the plan proposal under evaluation, are not achieved as a result of AmI environment changes.

The input of this process is $\mathsf{Ref}(\Pi, \Phi)$, the set of plans proposed by the agents at the previous plan proposal stage. Since ambient agents may have different available context information (represented as a combination of facts and defeasible rules) depending on their information sources, they may not agree on the evaluation of a plan proposal at some point during the dialogue. The EVALUATION stage generates as many argumentative dialogues as argument steps are present in the proposal plan under evaluation. An argumentative dialogue is an exchange of arguments for or against the fulfillment of an argument step, represented as a Plan Argument Dialogue (PAD) tree $\mathcal{T}_{\Pi_r}^{\mathcal{A}}$, where $\Pi_r \in \mathsf{Ref}(\Pi, \Phi)$ is the refinement plan to be evaluated and $\mathcal{A} \in \mathcal{AR}(\Pi_r)$ is the particular argument step to be evaluated. We denote the nodes in a PAD tree as tuples of the form $(\Pi_r, \mathcal{A}, \Gamma)$, where Γ is a set of attacking arguments (whose bases are warranted in the plan Π_r) that will finally determine if argument $\mathcal{A}$ is warranted in plan Π_r. Every node in a PAD tree (except the root) represents a defeater of its parent, and the leaves of the tree correspond to undefeated plans. The set of direct successors nodes of a given node Π_r, is denoted as $\mathsf{succ}(\Pi_r)$. More specifically:

1. The root of the tree is labeled with $(\Pi_r, \mathcal{A}, \emptyset)$.
2. A plan node $(\Pi_r, \mathcal{A}, \{\mathcal{B}\}) \in \mathsf{succ}(\Pi_r)$ represents an attack against the argument $\mathcal{A}$ in plan Π_r through the inclusion of an attacking argument, namely $\mathcal{B}$. Consequently, each node in $\mathsf{succ}(\Pi_r)$ stands for a defeater of the root argument $\mathcal{A}$, i.e. $\mathcal{B}$ is a defeater of $\mathcal{A}$.
3. A plan node $(\Pi_r, \mathcal{A}, \{\mathcal{B}, \mathcal{C}\}) \in \mathsf{succ}(\mathsf{succ}(\Pi_r))$ indicates an attack to the argument child $\mathcal{B}$ of the parent node through the inclusion of a new attacking argument, say $\mathcal{C}$, so this new node is a supporter of the root argument $\mathcal{A}$.

Informally we might see a PAD tree for an argument step $\mathcal{A}$ as generating a dialectical tree [5] for $\mathcal{A}$. But in DeLP-MAPOP the nodes in the PAD tree are contextualized within a plan. Every linear path from the root to a leaf corresponds to one different acceptable argumentation line. Circular argumentation (also known as fallacious argumentation) is avoided by applying both conditions from [5]: no argument can be reintroduced in the same argumentation line and argument concordance must be guaranteed.

Let Ag_2 be an agent that has the defeasible rules detailed in Listing 3, and $\{\langle$*device-measure-the-traffic-H1-pH, high* $\rangle$, $\langle$ *maps-google-distance-H1-pH, long* $\rangle\} \subseteq \Psi_{\mathsf{Ag}_2}$. When Ag_1 sends the PAD tree $\mathcal{T}_{\Pi_r}^{\mathcal{A}^{\mathsf{Ag}_1}}$ (containing only the root node) to the rest of agents, Ag_2 puts forward an attacking argument $\mathcal{B}^{\mathsf{Ag}_2} = (\{\langle$ *pos-t11, H1* $\rangle\}, \{\delta_0; \delta_1; \delta_2\})$, inspired by Listing 3, where:

- δ_0 =*(and ⟨ pos-t11, H1 ⟩ ⟨ at-a11, H1 ⟩)* $\prec$ *(and ⟨ moved-amb-a11-H1, pH ⟩ ⟨ moved-team-t11-H1, pH ⟩ ⟨ traffic-jam-between-H1-pH, true ⟩ ⟨ is-far-from-H1-pH, true ⟩).*
- δ_1 =⟨ *traffic-jam-between-H1-pH, true* ⟩ $\prec$⟨ *device-measure-the-traffic-H1-pH, high* ⟩.
- δ_2 =⟨ *is-far-from-H1-pH, true* ⟩ $\prec$ ⟨ *maps-google-distance-H1-pH, long* ⟩.

which attacks $\mathcal{A}^{\mathsf{Ag}_1}$. Unlike agent Ag_1, agent Ag_2 knows that traffic jam is expected according to a smart device from the AmI system that monitors the traffic density between the hospital *H1* and the patient's home *pH*, and also knows that the distance between them provided by a web mapping service as Google Maps, is rather large. Both informations may be a reason to believe that an ambulance, initially located at the hospital *H1* will not arrive to *pH* in time for assisting the patient. Thus, Ag_2 creates a new node $(\Pi_r^{\mathsf{Ag}_1}, \mathcal{A}^{\mathsf{Ag}_1}, \{\mathcal{B}^{\mathsf{Ag}_2}\}) \in \mathsf{succ}(\Pi_r^{\mathsf{Ag}_1})$ among others, and sends it to rest of agents.

```
(:def-rule moved-medical-assistance-denied
    :parameters(?a - ambulance
    ?a1 address-hospital ?a2 - address-patient-home
    ?m - medical-team)
    :head (and (assign (at ?a) ?a1)
               (assign (pos ?m) ?a1))
    :body (and (= (moved-amb ?a ?a1) ?a2)
               (= (moved-team ?m ?a1) ?a2)
               (= (traffic-jam-between ?a1 ?a2) true)
               (= (is-far-from ?a1 ?a2) true)))
(:def-rule traffic-jam
    :parameters(?a1 address-hospital
    ?a2 - address-patient-home)
    :head (assign (traffic-jam-between ?a1 ?a2) true)
    :body (= (device-measure-the-traffic ?a1 ?a2) high))
(:def-rule distance
    :parameters(?a1 address-hospital
    ?a2 - address-patient-home)
    :head (assign (is-far-from ?a1 ?a2) true)
    :body (= (maps-google-distance ?a1 ?a2) long))
```

Listing 3: Defeasible rules for representing situations in which the ambulance may not arrive on time.

In the next round of the dialogue, $(\Pi_r^{\mathsf{Ag}_1}, \mathcal{A}^{\mathsf{Ag}_1}, \{\mathcal{B}^{\mathsf{Ag}_2}\})$ is received by the ambient agent Ag_3 who discovers a new attacking argument $\mathcal{C}^{\mathsf{Ag}_3}$ that defeats $\mathcal{B}^{\mathsf{Ag}_2}$, which is based on Listing 4.

```
(:def-rule carpool-lane
    :parameters(?a1 address-hospital
    ?a2 - address-patient-home)
    :head (assign (traffic-jam-between ?a1 ?a2) false)
    :body (= (carpool-lane-between ?a1 ?a2) true))
```

Listing 4: The defeasible rule used for representing a carpool lane which may prevent an ambulance from being stuck by a traffic congestion situation.

Assuming that ⟨ *carpool-lane-between-H1-pH, true* ⟩ $\in \Psi_{\mathsf{Ag}_3}$, then $\mathcal{C}^{\mathsf{Ag}_3}$ = ({⟨ *traffic-jam-between-H1-pH, false* ⟩},{⟨ *traffic-jam-between-H1-pH, false* ⟩ $\prec$ ⟨ *carpool-lane-between-H1-pH, true* ⟩}). That is, Ag_3 knows that there is a carpool lane (as an express lane) between *H1* and *pH*, which is a reason to believe that the ambulance *a11* can skip the traffic congestion on the way to reach the patient's home. Ag_3 creates a new plan $(\Pi_r^{\mathsf{Ag}_1}, \mathcal{A}^{\mathsf{Ag}_1}, \{\mathcal{B}^{\mathsf{Ag}_2}, \mathcal{C}^{\mathsf{Ag}_3}\})$ extending $(\Pi_r^{\mathsf{Ag}_1}, \mathcal{A}^{\mathsf{Ag}_1}, \{\mathcal{B}^{\mathsf{Ag}_2}\})$ with $\mathcal{C}^{\mathsf{Ag}_3}$, and sends it to the rest of agents. The evaluation dialogue for $\mathcal{T}_{\Pi_r}^{\mathcal{A}^{\mathsf{Ag}_1}}$ continues until all defeaters are put forward in a round.

In order to check whether the argument of the root node is defeated or undefeated, the following procedure on the PAD tree is applied: label with a *U* (for *undefeated*) each terminal plan in the tree (i.e. each plan with no defeaters at all). Then, in a bottom-up fashion, we label a node with: *U* if each of its successors is labeled with a *D*; and *D* (for defeated) otherwise.

A plan in $\mathsf{Ref}(\Pi, \Phi)$ is labeled as an **undefeated refinement plan** if all the root plans of its PAD trees are labeled as undefeated. Otherwise the plan is provisionally labeled as a **defeated refinement plan** in the POP tree. Undefeated plans are obviously preferred over defeated plans as they represent a plan with no expectation failures according to the ambient agents. Nevertheless, defeated plans are maintained in the POP tree as their arguments may become later undefeated as the problem evolves and information changes. Finally, each ambient agent updates its initial facts and defeasible rules with the facts and defeasible rules from the exchanged arguments' bases.

3.4 Plan selection process

At the SELECTION stage, the aim is to select the next plan Π to be refined and continue with the plan-space planning process of the PROPOSALS stage, unless Π is already a solution in which case the DeLP-MAPOP protocol stops.

For selecting a plan, agents apply three criteria in order of priority over the set of evaluated plans from the previous stage. The objective is to select a plan considering a compromise between the desire to minimize the computational overhead and that of maximizing the quality of the solution plan. The three criteria are: first, the system applies a warranty procedure to discard the plans evaluated as defeated in the evaluation stage. Second, a heuristic function is applied over the undefeated plans resulting from the above filtering. We use two of the most popular heuristics in planning: SUM and MAX heuristics [16]. The SUM heuristic estimates the cost of a plan as the sum of the cost of the pending open goals in the plan whereas the MAX heuristic returns the value of the most costly open goal as heuristic estimation. Plans whose heuristic estimation is below a certain threshold are discarded from consideration. Finally, the last filtering over the remaining plans considers the preference functions. We have implemented two intersection techniques aimed at selecting the most preferable plan by the ambient agents according to their preferences. The first mechanism selects the plan whose actions are all among the preferences of every agent with a degree of preference above a certain threshold. If the application of this method returns an empty list then we compute the number of preferred actions in each plan and we select the plan with the largest proportion of preferred actions by the ambient agents.

4. EXPERIMENTAL EVALUATION

The purpose of this section is to test the overall performance, scalability and quality of DeLP-MAPOP versus a MAP system with no argumentation (MAPOP) which has also been implemented in the same agent platform, and discuss the benefits and limitations of each system. We carried out several experiments considering three different levels of difficulty of the planning problem: small (composed by 8 grounded actions and 50 grounded defeasible rules), medium (composed by 16 grounded actions and 100 grounded defeasible rules) and large (composed by 24 grounded actions and 150 grounded defeasible rules). We used teams of agents of different size ranging from 1 (single-agent) to 5. We performed several tests varying the number of agents of each type in the AmI environment, namely transportation, communication and assistant agents, and we took the median values over 20 repetitions for each set of experiments with 'n' agents, regardless the type of agent. We used the MAX heuristic and the Intersection function.

DeLP-MAPOP and MAPOP are implemented on Magentix2[2], a multi-agent platform based on Apache Qpid[3], an open-source implementation of Advanced Message Queuing Protocol for communication.

With regard to scalability and performance, Figures 2(a), 2(b) and 2(c) show the average time spent on each stage of the DeLP-MAPOP protocol, while Figure 2(d) shows the average total time to find a solution plan, including parsing the problem file and grounding the planning actions and defeasible rules. The horizontal axis (the same for the rest of the figures) depicts the size of the team of ambient agents, while the vertical axis displays the time in milliseconds. As expected, the average time spent in DeLP-MAPOP is always greater than the time spent in MAPOP due to the following reasons: i) in the PROPOSALS stage, the ambient agents from DeLP-MAPOP do not only have to reason about which actions would achieve the selected open goal, but also need to reason about which arguments would support it; ii) the EVALUATION stage is not considered in MAPOP; and iii) the SELECTION stage is replaced in MAPOP by a single heuristic function. It is also noticeable that the more agents in a team, the more exchanged messages between them, causing each stage to take longer in DeLP-MAPOP. Figure 2(e) illustrates precisely that, as the number of agents increases, the number of exchanged messages is larger; Figure 2(f) shows that as the size of the team increases, the number of dialogue rounds is lower because in this case more attacking arguments tend to appear in a single round, thus decreasing the number of rounds.

Figure 3 shows the evaluation of the quality of the obtained solution plans. Figure 3(g) shows that the average number of action steps in solution plans of DeLP-MAPOP is lower or equal than the average number in solution plans of MAPOP. The reason is that in MAPOP, an open goal that is not a threat can only be achieved through an action step, while in DeLP-MAPOP the open goal can also be supported by an argument step whose base is already guaranteed in the plan. In these cases, the cost of the DeLP-MAPOP plans is smaller because fewer actions are required to support the open goals, meaning that the agents' beliefs support the fulfillment of a goal without explicitly including an additional action in the plan. The fact that argument steps are not used in MAPOP is precisely shown in Figure 3(h). On the other hand, we can see in Figure 3(i) a comparison of the quality of plans generated with a single-agent team versus plans generated by teams with more than one agent. Obviously, in the first case, plans are sequential while DeLP-MAPOP returns plans with parallel actions that can be simultaneously executed by different ambient agents.

We also carried out one more experiment: which action steps in MAPOP solution plans are actually discarded during an argumentative dialogue in DeLP-MAPOP plans. This latter aspect is also a very relevant issue as we wanted to compare the plans returned by both systems and see how many plans, and actions correspondingly, of MAPOP were actually discarded by the agents in DeLP-MAPOP during the argumentative dialogues. The results of this experiment are shown in Figure 3(j). As can be seen, according to the knowledge of the ambient agents, 0% of the solution plans generated by DeLP-MAPOP comprise failing actions, i.e. actions whose intended effects were acknowledged to fail at the EVALUATION stage. Obviously, as long as agents acquire more information from the context, argumentative dialogues will fit reality better and, therefore, the guarantee of a successful solution plan (a plan with no expected failures) would also be greater. Furthermore, this experiment allowed us to check the correctness of the argumentative dialogues at the EVALUATION stage. However, in the case of MAPOP, up to 50% of the plans had actions that were discarded by the ambient agents in DeLP-MAPOP, that is, actions that agents acknowledged that would not be successfully executed.

Examining the influence of preferences in DeLP-MAPOP, Figure 3(k) shows that the average satisfaction of each team with the solution plans decreases as the size of the team increases. We calculated the satisfaction of an individual agent on a solution plan by averaging its preferences in the action steps of the plan, while the team satisfaction is calculated as the average of the individual satisfactions. Figure 3(l) shows that the difference of satisfaction between agents tends to increase as the size of the teams also increases. It is desirable that the difference is as small as possible for that all agents are equally satisfied.

5. CONCLUSIONS AND FUTURE WORK

This paper presents the specification, implementation and an exhaustive experimentation of DeLP-MAPOP, an argumentation-based defeasible planning framework, on AmI applications. Our most relevant contribution is a fully implemented MAP framework that has been extensively tested in AmI environments. DeLP-MAPOP realizes three independent but cooperative processes to propose, criticize, defend and select alternative plan proposals. The results show two advantages of DeLP-MAPOP over a MAP process with no argumentation: (i) since each plan step of a plan proposal is collaboratively argued, DeLP-MAPOP returns plans whose actions are not likely to fail at execution time according to the information and beliefs of the ambient agents; and (ii) since open goals can also be supported by argument steps whose base is warranted with the facts of the plan, the context information and defeasible reasoning of agents provide a means to satisfy goals of the problem without an explicit inclusion of a planning action; this avoids considering unnecessary action steps thus reducing the total cost of the plan.

As future work, we intend to test the effectiveness and feasibility of DeLP-MAPOP in a hospital pilot program, as well as an extension to temporal defeasible argumentation for MAP [17]. Currently, we are working on the development of a more elaborated heuristic function that (i) analyzes the transitions between the values a state variable can take, and (ii) considers the experiences from the plan evaluation process (case-based argumentation) to predict the potential number of attacks that a plan can receive. We are also interested in studying the influence of the trust on the sources (devices) used by the ambient agents to acquire the context information as well as how a trust level determines the conflict resolution between attacking arguments. Finally, a comparison with other MAP approaches will be considered.

Acknowledgements

This work is mainly supported by the Spanish Ministry of Science and Education under the FPU Grant reference AP2009-1896 awarded to Sergio Pajares Ferrando, and projects, TIN2011-27652-C03-01, Consolider Ingenio 2010 CSD2007-00022, and PROMETEO/2008/051.

6. REFERENCES

[1] E. Aarts. Ambient intelligence. *Adaptive Hypermedia and Adaptive Web-Based Systems*, Springer, pp. 548–568, 2004.

[2] A. Bikakis and G. Antoniou. Distributed defeasible contextual reasoning in ambient computing. *Ambient Intelligence*, Springer, pp. 308–325, 2008.

[2] `http://www.gti-ia.upv.es/sma/tools/magentix2/index.php`

[3] `http://qpid.apache.org/`

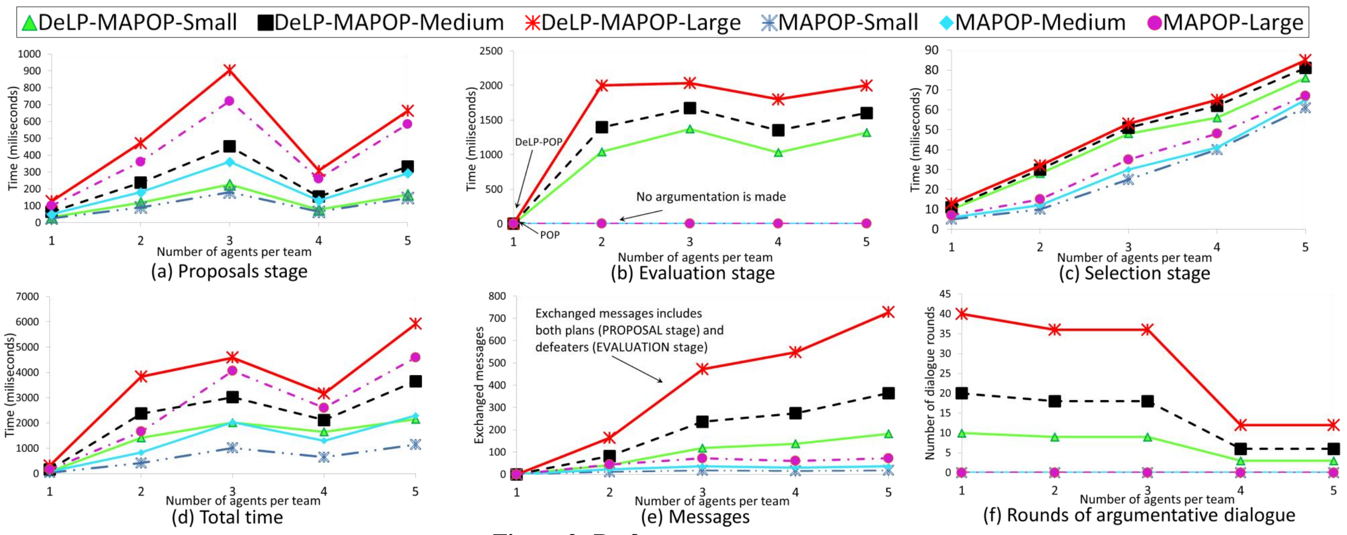


Figure 2: Performance measures.

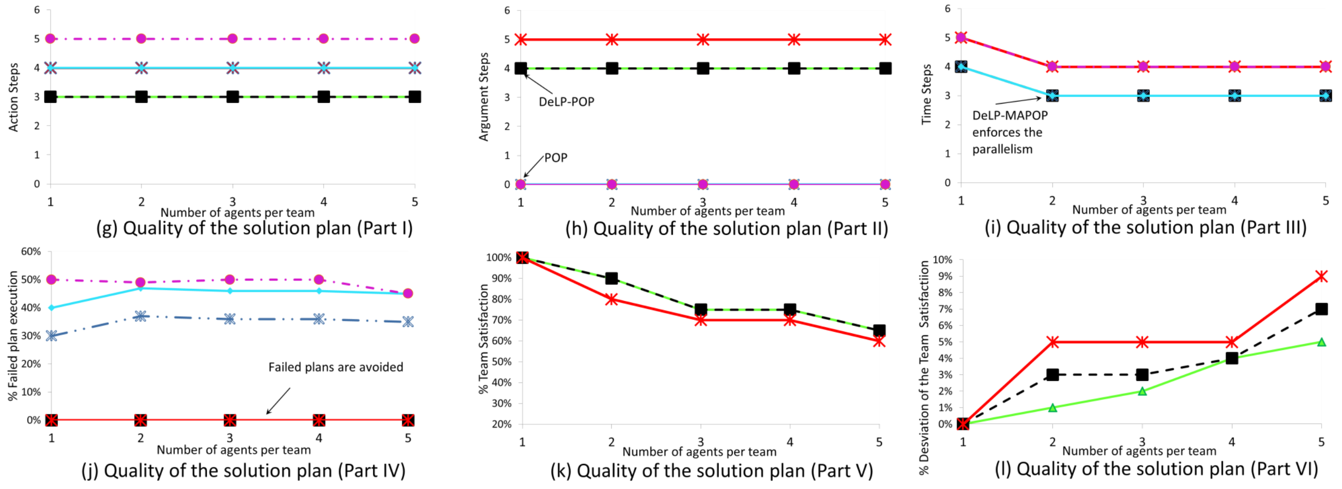


Figure 3: Quality measures.

[3] H. Prakken and G. Vreeswijk. Logics for defeasible argumentation. *Handbook of philosophical logic*, pp. 4:218–319, 2002.

[4] A. Bikakis and G. Antoniou. Defeasible contextual reasoning with arguments in ambient intelligence. *IEEE Transactions on Knowledge and Data Engineering*, pp. 1492–1506, 2010.

[5] A. García and G. Simari. Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming*, pp. 4:95–138, 2004.

[6] F. Amigoni, N. Gatti, C. Pinciroli and M. Roveri. What planner for ambient intelligence applications?. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, pp. 35(1):7–21, 2005.

[7] J.S. Penberthy and D.S. Weld. UCPOP: A Sound, Complete, Partial Order Planner for ADL. In Proc. of *KR*, pp. 103–114, 1992.

[8] D.E. Smith, J. Frank and A.K. Jónsson. Bridging the gap between planning and scheduling. *The Knowledge Engineering Review*, pp. 15(1):47–83, 2000.

[9] A. Jonsson, P. Morris, N. Muscettola, K. Rajan, and B. Smith. *Planning in interplanetary space: Theory and practice*. In Proc. of *ICAPS*, pp. 177–186, 2000.

[10] D. García, A. García, and G. Simari. Defeasible reasoning and partial order planning. In Proc. of the *International Conference on Foundations of information and knowledge systems, FoIKS 2008*, LNCS 4932, pp. 311–328, 2008.

[11] S. Pajares and E. Onaindía. Defeasible Planning through Multi-Agent Argumentation. *Modelling Machine Emotions For Realizing Intelligence, Smart Innovation Systems and Technologies Series*, pp. 13:311–342, 2011.

[12] P. Pardo, S. Pajares, E. Onaindía, L. Godo and P. Dellunde. Multiagent Argumentation for Cooperative Planning in DeLP-POP. In Proc. of *AAMAS*, pp. 971–978, 2011.

[13] S. Pajares, E. Onaindía and A. Torreño. An Architecture for Defeasible-Reasoning-Based Cooperative Distributed Planning. In Proc. of *CoopIS in conjunction with OTM*, pp. 200–217, 2011.

[14] M.L. Ginsberg and D.E. Smith. Reasoning about action II: The qualification problem. *Artificial Intelligence*, pp. 3:311–342, 1998.

[15] D.L. Kovacs. Complete BNF description of PDDL3.1. *Technical report*, 2011.

[16] X.L. Nguyen, and S. Kambhampati. Reviving partial order planning. In Proc. of *IJCAI*, pp. 17:459–466, 2001.

[17] S. Pajares and E. Onaindía. Temporal Defeasible Argumentation in Multi-Agent Planning. In Proc. of *IJCAI* , pp 2834–2835, 2011.