PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° MCC05/2017

Implementing an Argumentation-based Decision BDI Agent: a Case Study for Participatory Managment of Protected Areas

Pedro Elkind Velmovitsky

Jean Pierre Briot

Marx Viana

Carlos José Pereira de Lucena

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900

RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. MCC05/2017 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena March, 2017

Implementing na Argumentation-based Decision BDI Agent: a Case Study for Participatory Managment of Protected Areas

Pedro Elkind Velmovitsky, Jean Pierre Briot[1]
Marx Viana, Carlos José Pereira de Lucena

[1]Sorbonne Universités, UPMC Univ Paris 06, CNRS, Laboratoire d'Informatique de Paris 6 (LIP6), France

{pvelmovitsky, mleles, lucena}@inf.puc-rio.br, jean-pierre.briot@lip6.fr

Abstract. This paper describes the implementation of an argumentation system used for participatory management of environmental protected areas, more precisely to model the decision of a park manager artificial agent. This implementation is based on a BDI agent architecture, namely the Jason/*AgentSpeak* framework/language. After introducing the principles of BDI architecture and of argumentation systems, we will detail how we model arguments within the BDI (*Belief-Desire-Intention*) architecture. Then, we present the argumentation-based model of deliberation and decision by the park manager agent as a case study. We show how our argument-based approach allows to model various cognitive profiles of park managers (more conservationist or more sensitive to social issues), through different knowledge bases. We show examples of decisions produced by the park manager agent and examples of traces of arguments used during deliberation, which could be a base for explaining decisions. Before concluding, we point out future directions, such as using argumentation as a basis for negotiation between various agents.

Keywords: Agent architecture; BDI architecture; Argumentation; Decision; Participatory management.

Resumo. Este artigo descreve a implementação de um sistema de argumentação usado para gestão participatória de áreas ambientalmente protegidas, mais precisamente para modelar a decisão de um agente artificial representando o gestor de um parque. Essa implementação é baseada em uma arquitetura de agente BDI, nomeadamente o Jason/*AgentSpeak* framework/linguagem. Depois de introduzir princípios da arquitetura BDI e de sistemas de argumentação, iremos detalhar como modelar argumentos dentro da arquitetura BDI (*Belief-Desire-Intention/Crença-Desejo-Intenção*). Então, apresentaremos o modelo baseado em argumentação de deliberação e decisão do agente gestor do parque como um estudo de caso. Mostramos como nossa abordagem baseada em argumentação permite a modelagem de vários perfis cognitivos de gestores de parque (mais conservador ou mais sensível em relação com questões sociais), através de diferentes bases de conhecimento. Mostramos exemplos de decisões produzidas pelo agente gestor do parque e exemplos de traços de argumentos usados durante deliberação, e que podem servir de base para explicar decisões. Antes de concluir, apontamos direções futuras, como usar argumentação como base para negociação entre vários agentes.

Palavras-chave: Arquitetura de agente; Arquitetura BDI; Argumentação; Decisão; Gestão Participatória.

In charge of publications

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3114-1516 Fax: +55 21 3114-1530
E-mail: bib-di@inf.puc-rio.br

Table of Contents

1 Introduction

The general context of this work is an ongoing research project about exploring computer support for *participatory management* of protected areas (for biodiversity conservation and social inclusion). Therefore, we have designed a serious game (more precisely a distributed role-playing game), inspired by real management councils of national parks in Brazil (BRIOT, et al.). The applicative objective is to train people about participatory management of protected areas. More precisely, the idea is for players to explore *negotiation strategies* in order to address conflicting views. In this game, human players play various *roles* (representing members of a park management council, i.e., stakeholders, such as environmentalist, tourism operator, traditional population representative, etc.), discuss, negotiate and take *decisions* about *environment management*. In practice, these decisions are about the *type* (level) of *conservation* for each sub-part of the park. Example among predefined types, from more restricted conservation to more flexible kind, are: *Intangible* (full conservation), *Extensive* (flexible indirect use of resources), *Historic-Cultural* (possibility of limited tourism), *Conflicting* (shared identification of major conflicting opinions among council members). A special role is the *park manager*, who acts as an arbitrator in the game, making a final decision about the types of conservation for each part of the park and also being able to justify (or, at least, *explaining*) its decision to players.

This research project (BRIOT, et al.) explores the use of various advanced computer techniques as a support for *assistance* to the players. For instance, we introduce *expert* artificial agents helping players to analyze the *viability* of the consequences of their proposals, considering the *eco-socio-system dynamics* (in terms of biodiversity, economy, or/and social progress), and to compare them, as a support for objective comparison and negotiation (WEI, ALVAREZ and MARTIN, 2013). Another type of approach explored is using *argumentation* systems, as a support for *decision* and for *negotiation*, and last but not the least, for *explanation*. It could be used to model the decision of the park manager (or other kind/roles of artificial players), with an internal deliberation between arguments (BRIOT, et al.), as well as to support automated negotiation between players (artificial or human), as for instance as in (ADAMATTI, SICHMAN and COELHO, 2009). Using artificial players is useful to permit reproducible experiments with controllable levels of participation, as well as to compensate the absence of some human players.

In this paper, we will describe the modeling and implementation of an artificial agent playing the role of the park manager, based on an argumentation system (based on (RAHWAN and AMGOUD, 2006)), and modeled and implemented in a *BDI agent architecture*. BDI sounds for Belief-Desire-Intention and is one of the main cognitive architecture for agents and multi-agent systems (WOOLRIDGE, JENNINGS and KINNY, 1999). More precisely, we are using the *Jason* framework, an implementation of the *AgentSpeak* agent programming language, based on the BDI architecture (BORDINI, HUBNER and WOOLRIDGE, 2007).. Said another way, we are representing arguments and their management in *AgentSpeak*. The first version of the implementation has already been completed and tested. We will report on our experience as well as pointing out future directions.

The reminder of this paper is organized as follows: Section II discusses related work, while section III focuses on the background for BDI and for the Jason platform. Section IV presents the use of argumentation theory in BDI and Section V presents a case study for a BDI agent using argumentation. Section VI describes an implementa-

tion of the conflict resolution mechanism and Section VII shows its results. Finally, Section VIII presents our conclusions and future work.

2 Related Work

JogoMan-ViP (ADAMATTI, SICHMAN and COELHO, 2009) is a distributed role-playing game similar in spirit to our SimParc project, as its domain is about participatory management of hydric resources. It has introduced artificial players, also implemented through *AgentSpeak*/BDI on top of the Jason framework. This pioneering work has been an important influence. That said, the model of decision of artificial players is relatively simplified and with a predefined and fixed protocol for negotiation (to facilitate the interface between artificial players and human players). Our objective in using argumentation as the basis for decision, explanation and negotiation is even more ambitious.

Simulación by (GUYOT and HONIDEN, 2006) experimented with the use of artificial agents as assistants of human players, as an innovative and proactive type of interface. Assistant agents can make suggestions to human players, based on the model of a good strategy for the game combined with a learning mechanism. This work has also been an inspiration for our project. But we want to explore various models of the game, including *predictive* models (to estimate viability and resilience) and inner deliberation and justification models (by using *argumentation*).

Regarding argumentation systems, the theoretical framework by Rahwan and Amgoud (RAHWAN and AMGOUD, 2006) is interesting and a major source of inspiration, in that it is one of the first to use argumentation not only for *epistemic reasoning* (i.e., to create or modify knowledge) but also to *practical reasoning* (i.e., for reasoning about what to do and also how to do it).

(KAKAS and MORAITIS, 2003) is an implementation of an argumentation system based in Prolog. (PANISSON, et al., 2014) is a preliminary integration of an argumentation system into a BDI (*AgentSpeak*/Jason architecture), with similar objective to ours, but focusing on the implementation of defeasible logic (logical rules that can be refuted).

3 BDI and Jason

3.1 The BDI model

There are many ways to model the behavior of agents such as the BDI (Belief-Desire-Intention) model. To talk about this model, according to (BORDINI, HUBNER and WOOLRIDGE, 2007)., we need to address the idea that we can talk about computer programs as if they had a "*mental state*". Thus, when we talk about a belief−desire−intention system, we are talking about computer programs with computational analogues of beliefs, desires and intentions. These are described below:

Beliefs: are information the agent has about the environment. This information, however, is something the agent believes in but it may not be necessarily true. As an example, an agent may perceive from its environment the fact that it is raining. However, the rain may stop before the next reasoning cycle of the agent – in this case, his belief is outdated and incorrect.

Desires: are the possible states of affairs that the agent might like to accomplish. That does not mean, however, that the agent will act upon it – it is a potential influencer of the agent's actions.

Intentions: are the state of affairs that the agent has decided to act upon. In other words, intentions can be considered as a selected option between the potential set of options/desires that the agent has decided to pursue.

These are the key data structures of the BDI model. The decision-making approach used by the agent, therefore, is practical reasoning: the agent weighs conflicting information for and against the available options, according to its beliefs and desires. The result of this deliberation is the adoption of intentions, which in turn will lead the agent to execute actions.

3.2 Jason Platform

Jason is a Java-based interpreter for the *AgentSpeak* language (BORDINI, HUBNER and WOOLRIDGE, 2007)., providing a platform for the implementation and development of agents. This language is based on the BDI architecture systems (WOOLRIDGE, JENNINGS and KINNY, 1999) and allows programmers to customize the agent's beliefs, desires and intentions following logic sentences. As such, the agent architecture will have a *belief base*, which is changed and updated when the agent perceives information from its environment: new information may come to light, creating new beliefs, or old information may be found to be wrong or out of date, removing old beliefs. The agents also have goals that express the wishes the agent wants to accomplish. For instance, *!buy(car)* means that the agent has the goal of buying a car.

Furthermore, AgentSpeak provides a way to program and customize plans for the agents. These plans represent courses of action that the agents will take in order to achieve its goals. The overall syntax for a plan is:

triggering_event: context <- *body*.

Triggering events: represent changes which the agent will act upon – for instance, a change in its belief base or a change in its goals.

Context of a plan: is used to check the current situation so as to determine whether a particular plan among various alternative ones is likely to succeed in handling the event (e.g. achieving a goal), given the latest information the agent has about its environment (BORDINI, HUBNER and WOOLRIDGE, 2007)..

The body of a plan: is the course of action the agent will take in order to handle the event that triggered the plan.

4 Argumentation in BDI

There have been several theories which look at formalizing the reasoning of autonomous agents based on mental attitudes, such as beliefs, desires and intentions (BDI). One of the main characteristics of this type of reasoning is the resolution of *conflicts*, since the goals and attitudes available to the agent may not always be compatible. In addition, the information that the agent has may not always be consistent, or it may be true at one moment but incorrect on the next (RAHWAN and AMGOUD, 2006).

Argumentation is a promising approach to deal with such considerations. It is a mono-agent as well as a multi-agent process, in which an agent may decide alone, or adhere to the opinion of another agent, depending on the strength and validity of ar-

guments. Furthermore, agents reserve the right to *revisit* their *opinions* in light of new information.

The classical logic proves inadequate to model such behaviors — for example, to verify the property of monotony (SORDONI, et al., 2010):

If Φ, Δ and Γ denote sets of formulas in a formal reasoning system with the deduction :-, then the property of the monotony is stated in the following way:

$$If\ \Phi :- \Gamma\ and\ \Phi \subset \Delta\ then\ \Delta :- \Gamma$$

The addition of new formulas at Φ can never call into question the truth value of Γ. This is called a *closed world*.

The interest in non-monotonic logic appears when we try to capture the notion of everyday reasoning, where definite conclusions are obtained from incomplete information, which can be proven wrong or false and can be possibly correlated to the appearance of counterexamples. This is called an *open world* and *non-monotonous* reasoning: the addition of new formulas at Φ can call into question the truth of Γ.

However, this logical approach is limited to *epistemic reasoning* and does not modulate *practical reasoning*, limiting its use in agent architectures. So, a new approach has been developed to deal with non-monotonous reasoning, the argument.

As opposed to a proof, an argument may be invalidated. Moreover, by comparing arguments it is possible to manage inconsistencies in the agent's belief base.

In order to formalize the notion of an argument, we refer to concepts used by Rawhan and Amgoud in their work (RAHWAN and AMGOUD, 2006), and in turn, by our SimParc project (SORDONI, et al., 2010):

Let Λ be a logical language and Σ a knowledge base. The classical deduction is denoted by :- and the logical equivalence ≡. Then, an argument is a pair ⟨H, h⟩, $H \subseteq \Sigma$ such that:

- *H is consistent;*
- *H :- h;*
- *H is minimal, that is, there is no subset of H which verifies 1 and 2.*

From this definition, the authors define the attack relations between arguments:

Let A (Σ) denote the set of arguments that can be constructed from Σ. Let ⟨H1, h1⟩ and ⟨H2, h2⟩ two arguments belonging to A(Σ):

- *⟨H1, h1⟩ refute ⟨H2, h2⟩ iff h1 ≡ ¬h2;*
- *⟨H1, h1⟩ block ⟨H2, h2⟩ iff ∃h ∈ H2, h ≡ ¬h1.*

To represent players in the game, with their respective roles, the SimParc project utilizes the concepts above to model their agents. So, let us note *D* as the set of desires, *B* the set of beliefs and *A* the set of actions. Let's suppose, for example, that:

B = {road, tourism_flow, beach} and

A = {extensive_use, intangible_use}.

Each agent, then, will have the following rules and bases:

- The *rules to generate desires* (later on, named *desires rules) RD_i* have the form: φ *-:* $\beta_1, ..., \beta_m, \varphi_1, ..., \varphi_n, \beta_i \in B$ *and* $\varphi_i \in D$. If the agent has beliefs $\beta_1, ..., \beta_m$ and desires $\varphi_1, ..., \varphi_n$ then desire φ is satisfied. These rules belong to the base $BD = \{(RD_i, w_i)\}$, where RD_i represents a rule of desire and w_i represents the intensity (weight) of the conclusion desire φ. An example is:

+*road : tourism_flow & beach* <-+*raise_tourism(3),* where *(3)* means that *Intensity (raise_tourism) = 3*.

- The *decision rules RAj* have the form φ -: *a*, where $a \in A$ and $\varphi \in D$. If the agent takes the decision of performing action *a*, then desire φ is satisfied. These rules belong to the base $BA = \{(RAj, u_j\}$, where *RAj* represents a rule of desire and u_j represents the utility of the action *a* according to the desire φ. An example is:

  +*extensive_use(0.75)* -> +*raise_tourism(0.75), with Utilityraise_tourism (extensive_use) = 0.75.*

The + sign in the beginning of each rule in the examples means, in *AgentSpeak*, that the plans will be executed when the agent gains the respective belief.

There are also two more types of rules and associated knowledge bases: the epistemic rules, that generate new beliefs from older beliefs, and the viability rules, that determine the viability of actions according to the existing beliefs. In the simplified scenario presented here, we purposely consider that there are no additional beliefs to be inferred by the environment, and that all the actions defined are viable, so we omitted these rules. For more information on these rules, see (SORDONI, et al., 2010).

It is important to note that, for an agent to decide which action to choose, he must compare the gain of each action. These gains are defined as:

$gain_{\varphi}(a) = intensity(\varphi) * u_{\varphi}(a)$.

If an action is related to more than one desire, $\varphi_1, ..., \varphi_n$, than a gain vector of length *n* will be created with its values corresponding to each gain $\varphi_i(a)$. An aggregation function is used to transform the vector into a mathematical value, so that the gains of each action may be compared (see more details in (SORDONI, et al., 2010)).

5 Manager Agent in SimParc

5.1 SimParc and the Park Manager Role

The SimParc project focuses on participatory management of protected areas. More specifically, the protected areas modelled in the game are Brazilian national parks. The law defines the existence of a *management council*, only consultative in the case of national parks, to include social representation (*stakeholders*). The game models the negotiation process between the stakeholders, such as the park manager, representatives of local communities, NGOs, tourism operators, among others.

In the game, each stakeholder is played by a human with the goal of deciding how to best utilize the landscape units in the park. For instance, NGOs probably want more conservation of the areas, while tourism operators want more flexibility (e.g., for tourism visitation). They discuss and negotiate in order to try to achieve a common ground.

Each *landscape unit* represents a specific area of the park, with its own set of characteristics such as forests, roads, beaches and waterfalls.

As stated above, according to legislation about Brazilian natural parks, the only responsible to make final decisions concerning the use of landscape units is the *park manager*. But he will take into account the different opinions and interests of the stakeholders. Therefore, the focus of this work and paper is on the park manager role and the way to automate it into an artificial agent.

5.2 Park Manager's BDI Architecture

Following the system outlined in Section III, the park manager's BDI architecture can be modelled with the respective knowledge bases. For example, suppose that a landscape unit has only two available actions *extensive_use* and *intangible_use,* and the agent has the following beliefs about the unit: *{"road", "waterfall", "forest", "beach", "tourism_flow","forest_fire"}*. Then, desires rules (*RD* base) are modeled as shown in Table I.

For instance, RD_1 will add the desire *raise_tourism* with intensity *3*, if the agent believes the environment has a *tourism_flow* and a *beach* (the context). Table II shows the decision rules (RA base).

Name	Rule
RD_1	*+road : tourism_flow & beach <-+raise_tourism(3)*
RD_2	*+waterfall : true <- +~raise_tourism(2)*
RD_3	*+forest : true <- + protect_forest(3)*
RD_4	*+forest_fire: true <- +prevent_fire(3)*
RD_5	*+beach : ~protect_forest(A) <- +protect_forest_argument(C)*

Table 1. Desire rules (RD Base)

Table III shows the *Plans*, that are executed when the actions *extensive_use* and *intangible_use* become available. The *raise_tourism* desire related to *extensive_use* has utility *0.75*, the *protect_forest* and *prevent_fire* desires related to *intangible_us*e have utilities *1.0* and *0.5*, respectively.

Name	Rule
RA_1	*+extensive_use(0.75) : true <- !raise_tourism_ extensive_ use (0.75)*
RA_2	*+intangible_use(1.0) : true <- !protect_forest_intangible_use (1.0);!prevent_fire_intangible_use(1.0/2);!protect_forest_prevent _fire_intangible_use(1.0)*

Table 2. Decision rules (RA Base)

It is important to note that the values of *intensity* and *utility* are modelled by the programmer, according to the manager's "personality". If this particular park manager is more *socioconservationist*, he may prefer a more extensive use of the landscape unit and therefore the utility from the desire *raise_tourism* may be bigger. The same applies if the park manager is more *preservationist* and therefore prefers a more intangible use of the landscape unit.

With the rules and parameters mentioned above, it is important to note first that there are two attack relations between explanatory arguments:

- Refute attack between RD_1 and RD_2: since RD_1's conclusion is *raise_tourism*, and RD_2's conclusion is *~raise_tourism*, a refute attack relation is constituted. The in-

tensity of the desire in RD_1 is bigger than the intensity of the desire in RD_2, thus the *raise_tourism* desire continues on the agent's base.

- Block attack between RD_3 and RD_5: since RD_3's conclusion *protect_forest* is present in the body of RD_5, a block attack relation is constituted. The intensity of the desire in RD_5 is bigger than the intensity of the desire in RD_3, thus the desire *protect_forest* is removed from the agent's base.

With the *protect_forest* desire removed, its gain is *0*. So, the *extensive_use* action has a gain *3*0.5 = 1.5* for the *raise_tourism* desire. The *intangible_use* action has a gain *3*0.75 = 2.25* for the *prevent_fire* desire. Since *2.25* is bigger than *1.5*, the *extensive_use* action is selected.

Name	Plan
P_1	*+!raise_tourism_extensive_use(T): raise_tourism(A) <- .print (" extensive_use action added from desire raise_tourism, with utility ",D, " and total gain ", D*T)*
P_2	*+!raise_tourism_extensive_use(T) : not raise_tourism(A) <- .print ("extensive_use action added from desire raise_tourism, with utility ",D, " and total gain ", 0)*
P_3	*+protect_forest_argument(C) : protect_forest(A) & C>A <- .print ("protect_forest negation added with intensity ",C, " and desire protect_forest with intensity ", A, " removed");-protect_forest(A)*
P_4	*+protect_forest_argument(C) : protect_forest(A) & C<A <- .print ("protect_forest negation added with intensity ",C, " and desire protect_forest has bigger force ", A)*
P_5	*+~raise_tourism(A): raise_tourism(B) & A>B <- .print("raise_tourism negation added with intensity ", A, " and desire raise_tourism with intensity ",B, " removed");-raise_tourism(B)*
P_6	*+~raise_tourism(A): raise_tourism(B) & A<B <- .print("raise_tourism negation added with intensity ", A, " and desire raise_tourism with intensity ",B, " has bigger force")*
P_7	*+!protect_forest_prevent_fire_intangible_use(A) : not protect_ forest(E) & prevent_fire(D) & raise_tourism(C) & extensive_use (B) & C*B<=(D*A/2) <- .print("intangible_use action executed with total gain: ",(D*A/2),", because ", (D*A/2), " is bigger than ", C*B)*
P_8	*+!protect_forest_prevent_fire_intangible_use(A) : not protect_ forest(E) & prevent_fire(D) & raise_tourism(C) & extensive_use (B) & C*B>(D*A/2) <- .print("extensive_use action executed with total gain: ",C*B,", because ", C*B , " is bigger than ", (D*A/2))*

Table 3. Plans

6 Implementation of the Manager Agent in Jason

The park manager agent BDI architecture mentioned in section IV.B, with its respective *RD* and *RA bases*, has been implemented utilizing the Jason plug-in for *Eclipse*. Then, an implementation of the mechanism to compare arguments in order to eliminate conflicts has been modeled in this architecture.

For instance,*!raise_tourism_extensive_use(T)* is used to check if the *raise_tourism* desire was added from the *RD* in P_1 and P_2: since the agent has a rule that adds the *raise_tourism* desire, P_1 will be executed. If the agent's belief base did not include a *road*, for instance, then the *raise_tourism* desire would not be added from RD_1, and the agent would not believe that this desire is feasible – that is what the *not raise_tourism(A)* clause means. In this case, P_2 would be executed.

The goals *!protect_forest_intangible_use(T)* and *!prevent_ fire_intangible_use(T)* are used in a similar fashion, and are not described here.

It is important to note the difference between the *not l* and the *~l* operators in *AgentSpeak*: in the latter, the agent believes *l* is false, while in the former the agent does not believe *l* is true – which is not the same as believing *l* is false; the agent may be simply ignorant about *l*. Also, the - operator, used in P_3 and P_5, is used to remove the desire from the agent's base (BORDINI, HUBNER and WOOLRIDGE, 2007).

The *protect_forest_argument(C)* goal is used to check if a block attack relation exists in P_3 and P_4: in P_3, if the intensity of the negation of the *protect_forest desire*, denoted by *C*, is bigger than the intensity of the desire itself, denoted by *A*, then the *protect_forest* desire is removed. In P_4, if *C* is smaller than *A*, then the desire is not removed. In this implementation, *C* has been set to *4* and *A* to *3*, so P_3 applies and the desire is removed.

The refute attack relation, on the other hand, is tested in P_5 and P_6. In P_5, if the intensity of the negation of the *raise_tourism* desire, denoted by *A*, is bigger than the intensity of the desire itself, denoted by *B*, then the *raise_tourism* desire is removed. In P_6, if A is smaller than *B*, then the desire is not removed. In this implementation, *A* has been set to *2* and *B* to *3*. So, P_6 applies and the desire is not removed.

Lastly, the plan *!protect_forest_prevent_fire_intangible_ use(T)* is used to consolidate all results. Since this goal is part of the body of the plan of the *intangible_use* action, its context needs to test if the desires *protect_forest*, *prevent_fire* and *raise_tourism* are available and if the action *extensive_use* is available. Then, it checks the gains of each action accordingly and selects an action to perform. For instance, P_7 and P_8 test if the desire *protect_forest* is present in the belief base and if the other desires are still feasible, if the *extensive_use* action is still available and if the gains of the *intangible_use* action are bigger than the gains of the *extensive_use* action.

Similar clauses, omitted here, have been added for all possible combinations in the context, so that the agent will always know how to calculate the gains and choose an action.

7 Results

The results of the implementation in Jason, as logged by the agent (through the .print command), are:

- *extensive_use* action added from desire *raise_tourism*, with utility *0.75* and total gain *2.25*;
- *raise_tourism* negation added with intensity *2* and desire *raise_tourism* with intensity *3* has bigger force;
- *protect_forest* negation added with intensity *4* and desire *protect_forest* with intensity *3* removed;
- *intangible_use* action added from desire *prevent_fire* with utility *0.5* and total gain *1.5*;
- *extensive_use* action executed with total gain *2.25*, because 2.*25* is bigger than *1.5*;

Thus, the resulting decision by the park manager is *extensive_use.*

Suppose that we want to change the profile of the park manager from *socioconservationist* to *preservationist*, using the *protect_forest* desire as a parameter. Then, the intensity of ~*protect_forest* will be slightly adjusted from *4* to *2*, causing the desire *protect_forest* to not be removed in the attack relation. The total utility of the *intangible_use* action will be *3*0.5 (prevent_fire desire) + 3*1.0 (protect_forest desire) = 4.5*, surpassing the utility of the *extensive_use* action. The result, as traced by the agent, would be as following:

- *extensive_use* action added from desire *raise_ tourism*, with utility *0.75* and total gain *2.25*;
- *raise_tourism* negation added with intensity *2* and desire *raise_tourism* with intensity *3* has bigger force;
- *protect_forest* negation added with intensity *2* and desire *protect_forest* has bigger force *3*;
- *intangible_use* action added from desire *protect_forest* with utility *1* and total gain *3*;
- *intangible_use* action added from desire *prevent_fire* with utility *0.5* and total gain *1.5*;
- *intangible_use* action executed with total gain *4.5*, because *2.25* is bigger than *2.25*;

In this second case, the decision is thus *intangible_use*. This example shows the flexibility of the BDI model that makes use of the argumentation: it allows for the agent to deliberate and decide based on its beliefs and feasible desires while at the same time eliminating conflicts between them. In this case, the conflict between the *protect_forest* desire and its negation is resolved by comparing their respective forces.

8 Conclusion and Future Work

In this paper, we have presented a prototype architecture of an artificial agent able to make decisions by comparing arguments. The architecture is based on a BDI architecture (namely, the Jason/*AgentSpeak* framework/language). We have modeled an argumentation system through different knowledge base layers and the relation of attacks between arguments, as a basis to select best viable arguments. We have tested our architecture by modeling a park manager in a serious game for participatory management of protected areas. The park manager artificial agent makes decisions about conservation types by examining and reasoning (comparing) facts and arguments about the protected area situation and concerns.

As a future work, already ongoing, we plan to completely automate the management of attacks between arguments, in order to increase the agent's autonomy and reasoning capacity. In addition, being able to track the agent's reasoning to choose an argument over another enables the agent to provide the user feedback (online or offline, which could be used for explanation) about what is happening in the system. Last, argumentation could also be used, not only for internal deliberation within a single agent, but also for negotiation between agents by exchanging and evaluating arguments. This leads to the prospects of artificial negotiating players.

References

1. ADAMATTI, D. F.; SICHMAN, J. S.; COELHO, H. An analysis of the insertion of virtual players in GMABS methodology using the Vip-Jogoman prototype. **Journal of Artificial Societies and Social Simulation**, v. 12, n. 3, p. 7, 2009.
2. BARRETEAU, O. *et al*. Our companion modelling approach. **Journal of Artificial Societies and Social Simulation**, v. 6, n. 1, 2003.
3. BARRETEAU, O. *et al*. Role-playing games for opening the black box of multi-agent systems: method and lessons of its application to Senegal River Valley irrigated systems. 2001.
4. BORDINI, R. H.; HUBNER, J. F.; WOOLRIDGE, M. Programming multi-agent systems in AgentSpeak using Jason. **John Wiley & Sons**, 2007.
5. BRIOT, J.-P. *et al*. Participatory Management of Protected Areas for Biodiversity Conservation and Social Inclusion. In D. Adamatti (Ed), **Multi-Agent Based Simulations Applied to Biological and Environmental Systems, Advances in Computational Intelligence and Robotics (ACIR) Book Series.** IGI-Global, pp. 295–332.
6. DIGNUM, F. et al. Games and agents: Designing intelligent gameplay. **International Journal of Computer Games Technology**, v. 2009, 2009.
7. GUYOT, P.; HONIDEN, S. Agent-Based Participatory Simulations: Merging Multi-Agent Systems and Role-Playing Games. **Journal of Artificial Societies and Social Simulation**, V. 9, N. 4, 2006.
8. HOCINE, N.; GOUAICH, A.. A survey of agent programming and adaptive serious games. **RR-11013**, 2011, pp. 8. Disponível em http://hal-lirmm.ccsd.cnrs.fr/lirmm-00577722
9. KAKAS, A.; MORAITIS, P. Argumentation based decision making for autonomous agents. In: PROCEEDINGS OF THE SECOND INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS - ACM, 2003. p. 883-890. [
10. LE PAGE, C. *et al*. Participatory agent-based simulation for renewable resource management: the role of the Cormas simulation platform to nurture a community of practice**. Journal of Artificial Societies and Social Simulation**, v. 15, n. 1, p. 10, 2012.
11. PANISSON, A. R. *et al*. An Approach for Argumentation-based Reasoning Using Defeasible Logic in Multi-Agent Programming Languages. In: 11TH INTERNATIONAL WORKSHOP ON ARGUMENTATION IN MULTIAGENT SYSTEMS. 2014.
12. RAHWAN, I.; AMGOUD, L. An argumentation based approach for practical reasoning. In: PROCEEDINGS OF THE FIFTH INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTIAGENT SYSTEMS. ACM, 2006. p. 347-354.
13. SORDONI, A. et al. Design of a participatory decision making agent architecture based on argumentation and influence function – Application to a serious game about biodiversity conservation. **RAIRO – An International Journal on Operations Research**, p. 269–284. 2010.

14. WEI, W.; ALVAREZ, I.; MARTIN, S. Sustainability Analysis: Viability Concepts to Consider Transient and Asymptotical Dynamics in Socio-Ecological Tourism-based Systems. **Ecological Modelling,** V. 251, P. 103-113, 2013.

15. WOOLRIDGE, M.; JENNINGS, N. R.; KINNY, D. A methodology for agent-oriented analysis and design. In: PROCEEDINGS OF THE THIRD INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS AGENTS 99. 1999.