The hedgehog and the fox. An argumentation-based decision support system

Maxime Morge and Paolo Mancarella

via F. Buonarroti, 2
I-56127 Pisa, Italy
{morge,paolo}@di.unipi.it

Abstract. We present here a Decision Support System (DSS) with the help of a use-case for selecting a business location. This computer system is built upon a concrete argumentation framework for practical reasoning. A logic language is used as a concrete data structure for holding the statements like knowledge, goals, and actions. Different priorities are attached to these items corresponding to the likelihood of the knowledge, the preferences between goals, and the credibility of alternative actions. These concrete data structures consist of information providing the backbone of arguments. Due to the abductive nature of decision making, we built arguments by reasoning backwards. Moreover, arguments are defined as tree-like structures. In this way, our DSS evaluates the possible actions, suggests some solutions, and provides an interactive and intelligible explanation of the choice made.

1 Introduction

Decision making is the cognitive process leading to the selection of a course of action among alternatives based on estimates of the values of those alternatives. Indeed, when a human identifies her needs and specifies them with high-level and abstract terms, there should be a possibility to select some existing solutions. Decision Support Systems (DSS) are computer-based systems that support decision making activities including expert systems and multi-criteria decision analysis. However, these approaches are not suitable when the decision maker has partial and conflicting information. Further, standard decision theory provides little support in giving intelligible explanation of the choice made.

Since a decision can be resolved by confronting and evaluating the justifications of different positions, argumentation can support such a process. This is the reason why many works in the area of Artificial Intelligence focus on computational models of argumentation. In particular, nonmonotonic logic techniques have been used as a model with hierarchies of possibly conflicting rules (see [9] for a survey). However, even if modern techniques are used, this logical approach is still limited to the epistemic reasoning and do not encompass practical reasoning. The point is that a decision is not limited to draw conclusions. For this purpose, we consider practical reasoning as the vehicle of decision making, which is a knowledge-based, goal-oriented, and action-related reasoning. This kind of reasoning evaluates the state of affairs about some alternative courses of actions to achieve a set of goals.

In this paper, we present a Decision Support System (DSS) with the help of a use-case for selecting a business location. For this purpose, we propose an AF for practical reasoning. A logic language is used as a concrete data structure for holding the statements like knowledge, goals, actions. Different priorities are attached to these items corresponding to the likelihood of the knowledge, the preferences between goals, and the credibility of alternative actions. These concrete data structures consist of information providing the backbone of arguments. Due to the abductive nature of decision making, arguments are built by reasoning backwards. Moreover, arguments are defined as tree-like structures. In this way, our DSS evaluates the possible actions, suggests some solutions, and provides an interactive and intelligible explanation of the choice made.

Section 2 presents the principle and the architecture our DSS. Section 3 introduces the walk-through example. In order to present our Argumentation Framework (AF) for practical reasoning, we will browse the following fundamental notions. First, we define the *object language* (cf Section 4) and the priorities (cf Section 5). Second, we will focus on the internal structure of *arguments* (cf Section 6). We present in Section 7 the *interactions* amongst them. These relations allow us to give a declarative model-theoretic *semantics* to this framework (cf section 8) and we adopt a dialectical proof *procedure* to implement it (cf Section 9). Section 10 briefly discusses some related works. Section 11 concludes with some directions for future work.

2 Principle and architecture

Basically, human beings are categorized as either "hedgehogs", which know one big thing, or "foxes", which know many little things [3]. While most of the DSS are addressed to "hedgehogs", we want to provide one for both.

An "hedgehog" is an expert of a particular domain, who has intuitions and strong convictions. A "fox" is not an expert but she knows many different things in different domains. She decides by interacting with others and she is able to change her mind. Most of the DSS are addressed to "hedgehogs". These computer systems provide a way to express qualitative and/or quantitative judgements and show how to synthesize them in order to suggest some solutions. A decision taken with the help of a hedgehog could be great, but a full decision of hedgehogs could be a disaster. Since executives do not want to hear that a problem is complex and uncertain, decision makers need many hedgehog qualities. However the analytic skills needed for good judgments are those of foxes. We want to provide a DSS for the effective management of teams including both hedgehogs and foxes.

Figure 1 represents the current architecture of our DSS based upon an assistant agent. The mind of the agent relies upon MARGO (Multiattribute ARGumentation framework for Opinion explanation), i.e. our argumentative engine.

The hedgehog informs the assistant agent in order to structure and evaluate the decision making problem, by considering the different needs, by identify the alternative actions (alternatives, for short), and by gathering the required knowledge. As we will see in the next section, the agent uses concrete data structures for holding the hedgehog's knowledge, goals, and actions. These concrete data structures consist of information providing the backbone of arguments used to interact with the fox. The latter can ask for a possible solutions (*challenge*). MARGO suggests some solutions (*argue*). The reasons supporting these admissible solutions can be interactively explored (*challenge/argue*).

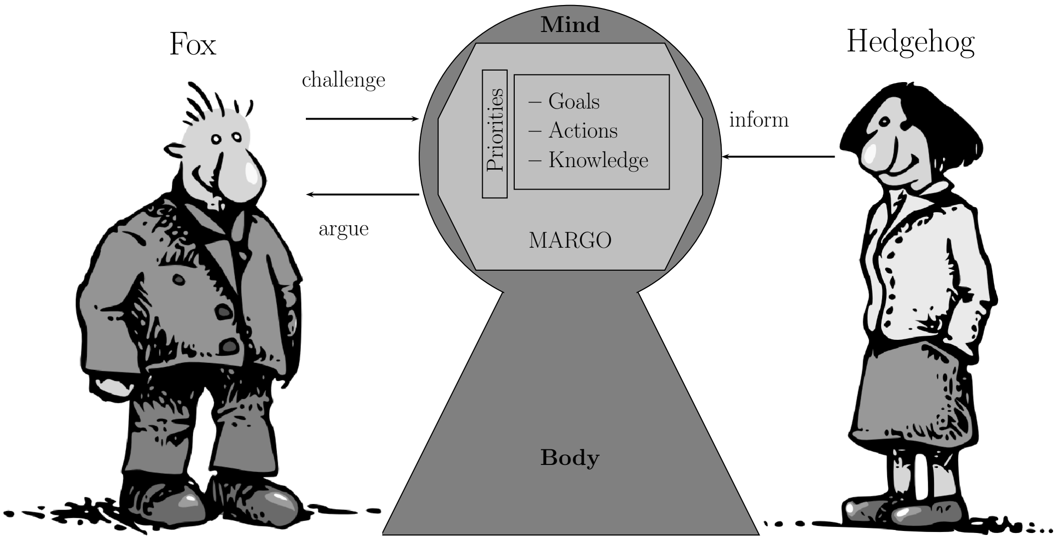

Fig. 1. Architecture of the argumentative agent

3 Walk-through example

We consider here the decision making problem for selecting a suitable business location.

In order to be successful, an investment requires a proper understanding of all relevant aspects. Detailed needs for the business location such as the government regulation, taxes, and so on as well as the knowledge about the quality of infrastructure and services, such as the availability of sea transports, is also of vital importance. The assistant agent is responsible for suggesting some suitable locations, based on the explicit users'needs and on their knowledge. The main goal, that consists in selecting the location, is addressed by a decision, i.e. a choice amongst some alternatives (Pisa or London). The main goal (g_0) is split into independent sub-goals and independent sub-goals of these sub-goals. The location must offer a "good" regulation (g_1) and a "great" accessibility (g_2). These high-level goals, which are *abstract*, reveal the user's needs. The location offers a "good" regulation, if the taxes are low (g_3), the permit can be easily obtained (g_4) and an assistance is available (g_5). These low-level goals are *concrete*, i.e. some criteria for evaluating different alternatives. The knowledge about the location is expressed with predicates such as: $\texttt{Sea}(x)$ (the location is accessible by sea transports), or $\texttt{Road}(x)$ (the location is accessible by road transports).

Figure 2 provides a simple graphical representation of the decision problem called influence diagram. The elements of the decision problem, i.e. *values* (represented by rectangles with rounded corners), *decisions* (represented by squares) and *knowledge* (represented by ovals), are connected by arcs where predecessors affect successors. We consider here a multiattribute decision problem captured by a hierarchy of values. The abstract values (represented by rectangles with rounded corner and double line) aggregate the values in the lower levels. When the structure of the decision is built, the alternatives must be identified, the priorities must be expressed and the knowledge gathered.

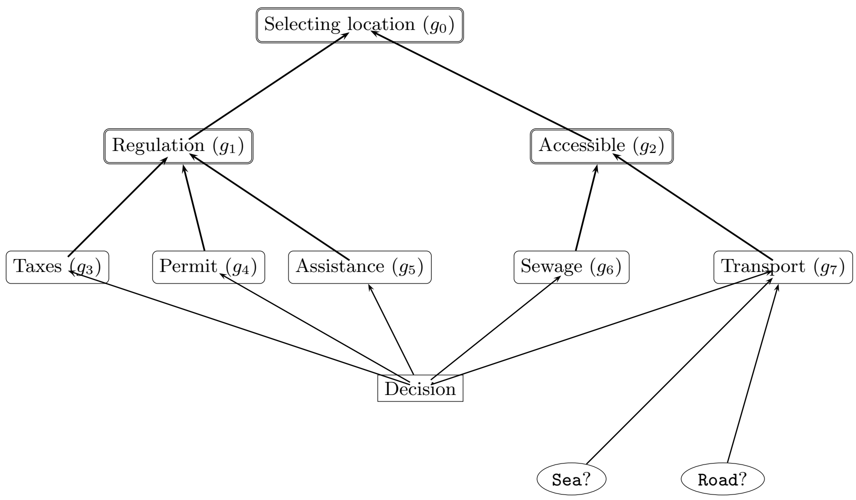

Fig. 2. Influence diagramm to structure the decision

While the influence diagram displays the structure of the decision, the object language and the priorities reveal the hidden details of the decision making.

4 The object language

Since we want to provide a computational model of argumentation and we want to instantiate it for our usecase, we need to specify a particular logic.

The object language, which is of logic-programming style, consists of two basic ingredients, i.e. a set of facts and a set of rules:

- a set of abstract *goals*, i.e. some propositional symbols which represent the abstract features that the decision must exhibit (in the example g_0, g_1, and g_2);
- a set of concrete *goals*, i.e. some propositional symbols which represent the concrete features that the decision must exhibit (in the example g_3, g_4, g_5, g_6, and g_7);
- a *decision*, i.e. a predicate symbol which represents the choice which must be made (in the example D);
- a set of *alternatives*, i.e. some constants symbols which represent the mutually exclusive alternatives for the decision (in the example `pisa` or `london`);
- a set of *beliefs*, i.e. some predicate symbols which represent epistemic statements (in the example `Sea` or `Road`).

Since we want to consider conflicts in this object language, we need some forms of negation. For this purpose, we only consider strong negation, also called explicit or classical negation[1]. A strong literal is an atomic first-order formula, possible preceded by strong negation $\neg$. $\neg L$ says "L is definitely not the case". Since we restrict ourselves to logic programs, we cannot express in a compact way the mutual exclusion between statements. For this purpose, we define the incompatibility relation (denoted by $\mathcal{I}$) as a binary relation over atomic formulas which is symmetric. For each atom L, we have $L \mathcal{I} \neg L$. Obviously, $D(a_1) \mathcal{I} D(a_2)$, D being a decision predicate, a_1 and a_2 being different alternatives[2] for D. Similarly, we say that a sentence ϕ_1 is incompatible with a set of sentences Φ_2 ($\phi_1 \mathcal{I} \Phi_2$) iff there is a sentence ϕ_2 in Φ_2 such that $\phi_1 \mathcal{I} \phi_2$ and we say that two sets of sentences Φ_1 and Φ_2 are incompatible ($\Phi_1 \mathcal{I} \Phi_2$) iff there is a sentence ϕ_1 in Φ_1 and a sentence ϕ_2 in Φ_2 such as $\phi_1 \mathcal{I} \phi_2$. A theory gathers the statements about the decision making problem.

Definition 1 (Theory). *A* theory $\mathcal{T}$ *is an extended logic program, i.e a finite set of rules of the form $R : L_0 \leftarrow L_1, \ldots, L_n$ with $n \geq 0$, each L_i being a strong literal. The literal L_0, called the* head *of the rule, is denoted by $head(R)$. The finite set $\{L_1, \ldots, L_n\}$, called the* body *of the rule, is denoted by $body(R)$. The body of a rule can be empty. In this case, the rule, called a* fact, *is an unconditional statement. R, called the name of the rule, is an atomic formula. All variables occurring in a rule are implicitly universally quantified over the whole rule. A rule with variables is a scheme standing for all its ground instances.*

Considering a decision making problem, we distinguish:

[1] Weak negation considered e.g. in [8] is not useful in our usecase.

[2] Notice that in general a decision can be addressed by more than two alternatives.

– *goal rules* of the form $R : g_0 \leftarrow g_1, \ldots, g_n$ with $n > 0$. Each g_i is a goal. According to this rule, the abstract goal g_0 is reached if the combination of goals in the body are reached;
– *epistemic rules* of the form $R : B_0 \leftarrow B_1, \ldots, B_n$ with $n \geq 0$. Each B_i is a belief literal. According to this rule, the belief B_0 is true if the conditions $B_1, \ldots, B_n$ are satisfied;
– *decision rules* of the form $R : g \leftarrow D(a), B_1, \ldots, B_n$ with $n \geq 0$. The head of the rule is a concrete goal and the body includes a decision literal ($D(a)$) and a possible empty set of belief literals. According to this rule, the goal can be eventually reached by the decision $D(a)$, provided that conditions $B_1, \ldots, B_n$ are satisfied.

Considering statements in the theory is not sufficient to take a decision.

5 Priority

In order to evaluate the previous statements, all relevant pieces of information should be taken into account, such as the likelihood of knowledge, the preferences between goals, or the credibility of the alternatives.

The decision maker can express these pieces of information by order relations. Actually, order relations are binary relations on a set. Since these relations classify the elements from the 'best' to the 'worst', with or without *ex æquo*, they are qualitative. For this purpose, we can consider either a preorder, i.e. a reflexive and transitive relation considering possible *ex æquo*, or an order, i.e. an antisymmetric preorder relation. The preorder (resp. the order) is total iff all elements are comparable. In this way, we consider that the *priority* $\mathcal{P}$ is a (partial or total) preorder on the rules in $\mathcal{T}$. $R_1\ \mathcal{P}\ R_2$ can be read "R_1 has priority over R_2". $R_1 \not\mathcal{P} R_2$ can be read "R_1 has no priority over R_2", either because R_1 and R_2 are *ex æquo* (denoted $R_1 \sim R_2$), i.e. $R_1\ \mathcal{P}\ R_2$ and $R_2\ \mathcal{P}\ R_1$, or because R_1 and R_2 are not comparable, i.e. $\neg(R_1\ \mathcal{P}\ R_2)$ and $\neg(R_2\ \mathcal{P}\ R_1)$.

In this work, we assume that all rules are potentially defeasible and that the priorities are extra-logical and a domain-specific feature. The priority of rules depends of the nature of rules and it is defined only for *concurrent* rules. Two rules R_1 and R_2 are *concurrent* rules iff either $\texttt{head}(R_1) = \texttt{head}(R_2)$ or $\texttt{head}(R_1)\ \mathcal{I}\ \texttt{head}(R_2)$. We define three priority relations:

– the priority over *goal rules* comes from their levels of preferences. Let us consider two concurrent rules R_1 and R_2. R_1 has priority over R_2 if the achievement of the goals in the body of R_1 are more important than the achievement of the goals in the body of R_2 ;
– the priority over *epistemic rules* comes from their levels of certainty. Let us consider two concurrent facts F_1 and F_2. F_1 has priority over F_2 if the first is more likely to hold than the second one;
– the priority over *decision rules* come from their levels of credibility. Let us consider two concurrent rules R_1 and R_2. R_1 has priority over R_2 if the first conditional decision is more credible than the second one.

In order to illustrate the notions previously introduced, let us go back to the decision making problem example. The goal theory, the epistemic theory, and the decision theory are represented in Table 1, Table 2, and Table 3 respectively. A rule above another one has priority over it if they are concurrent rules. To simplify the graphical representation of the theories, they are stratified in non-overlapping subsets, i.e. different levels. The *ex æquo* concurrent rules are arbitrarily regrouped in the same level. Non-comparable rules are arbitrary affected to a level[3]. According to the goal theory, the achievement of the goals

Table 1. The goal theory

$R_{012} : g_0 \leftarrow g_1, g_2$
$R_{1345} : g_1 \leftarrow g_3, g_4, g_5$
$R_{267} : g_2 \leftarrow g_6, g_7$
$R_{145} : g_1 \leftarrow g_4, g_5$
$R_{01} : g_0 \leftarrow g_1$
$R_{13} : g_1 \leftarrow g_3$
$R_{26} : g_2 \leftarrow g_6$
$R_{02} : g_0 \leftarrow g_2$
$R_{14} : g_1 \leftarrow g_4$
$R_{27} : g_2 \leftarrow g_7$
$R_{15} : g_1 \leftarrow g_5$

Table 2. The epistemic theory

$F_1 : \texttt{Road}(a_2) \leftarrow$
$F_2 : \texttt{Sea}(a_2) \leftarrow$
$F_3 : \neg\texttt{Road}(a_2) \leftarrow$

g_3, g_4 and g_5 is required to reach g_1 but this constraint can be relaxed and the achievement of g_4 is more important than the achievement of g_5 to reach g_1. According to the epistemic theory, the assistant agent does not know if London is accessible by sea/road transports. Due to conflicting sources of information, the agent has conflicting beliefs about the road accessibility of Pisa. Since these sources of information are more or less reliable, $F_1 \ \mathcal{P} \ F_3$. According to the decision theory, Pisa is more credible than London to reach g_3. The credibility of these alternatives with respect to g_7 depends on the knowledge: a location accessible by sea is more credible than a location accessible by road. We will build now arguments in order to compare the alternatives.

We will build now arguments in order to compare the alternatives.

[3] Notice that non-concurrent rules are not comparable, e.g. R_{1345} and R_{01}

Table 3. The decision theory

$R_{32} : g_3 \leftarrow D(\mathtt{pisa})$
$R_{41} : g_4 \leftarrow D(\mathtt{london})$
$R_{51} : g_5 \leftarrow D(\mathtt{london})$
$R_{71}(x) : g_7 \leftarrow D(x), \mathtt{Sea}(x)$
$R_{31} : g_3 \leftarrow D(\mathtt{london})$
$R_{42} : g_4 \leftarrow D(\mathtt{pisa})$
$R_{52} : g_5 \leftarrow D(\mathtt{pisa})$
$R_{61} : g_6 \leftarrow D(\mathtt{london})$
$R_{62} : g_6 \leftarrow D(\mathtt{pisa})$
$R_{72}(x) : g_7 \leftarrow D(x), \mathtt{Road}(x)$

6 Arguments

In this Section, we define and construct arguments by reasoning backwards due to the abductive nature of the decision making. Since we adopt a tree-like structure of arguments, our framework provides some intelligible explanations of alternatives.

The simplest way to define an argument is as a pair ⟨ premises, conclusion ⟩. This definition leaves implicit that the underlying logic validates a proof of the conclusion from the premises. When the argumentation framework is built upon an extended logic program, an argument is often defined as a sequence of rules [9]. These definitions ignore the recursive nature of arguments: arguments are composed of subarguments, subarguments for these subarguments, and so on. For this purpose, we adopt and extend the tree-like structure for arguments proposed in [11].

Definition 2 (Argument). *An* argument *is composed by a conclusion, a top rule, some premises, some hypotheses, and some sentences. These elements are abbreviated by the corresponding prefixes. An argument A is:*

1. *a* hypothetical argument *built upon an unconditional ground statement.*
   If L is a ground belief literal such that there is no rule R in $\mathcal{T}$ which can be instantiated in such a way that $L = \mathtt{head}(R)$, then the argument built upon this ground literal is defined as follows: $\mathtt{conc}(A) = L$, $\mathtt{top}(A) = \emptyset$, $\mathtt{premise}(A) = \emptyset$, $\mathtt{hyp}(A) = \{L\}$, $sent(A) = \{L\}$.
   or
2. *a* trivial argument *built upon an unconditional ground statement.*
   If F is a fact in $\mathcal{T}$, then the argument A built upon the ground instance F^g of F is defined as follows: $\mathtt{conc}(A) = \mathtt{head}(F^g)$, $\mathtt{top}(A) = F^g$, $\mathtt{premise}(A) = \{\mathtt{head}(F^g)\}$, $\mathtt{hyp}(A) = \emptyset$, $\mathtt{sent}(A) = \{\mathtt{head}(F^g)\}$.
   or
3. *a* tree argument *built upon an instantiated rule such that all the literals in the body are the conclusion of subarguments.*
   If R is a rule in $\mathcal{T}$, we define the argument A built upon a ground instance R^g

of R as follows. Let $\texttt{body}(R^g) = \{L_1, \ldots, L_n\}$ and $\texttt{sbarg}(A) = \{A_1, \ldots, A_n\}$ be a collection of arguments such that, for each $L_i \in \texttt{body}(R^g)$, $\texttt{conc}(A_i) = L_i$ (each A_i is called a subargument of A). Then: $\texttt{conc}(A) = \texttt{head}(R^g)$, $\texttt{top}(A) = R^g$, $\texttt{premise}(A) = \texttt{body}(R^g)$, $\texttt{hyp}(A) = \cup_{A' \in sbarg(A)} \texttt{hyp}(A')$, $\texttt{sent}(A) = \cup_{A' \in sbarg(A)} \texttt{sent}(A') \cup \texttt{body}(R^g)$.
A tree argument must be consistent, i.e. $\texttt{sent}(A)$ is neither incompatible with itself nor incompatible with $\texttt{conc}(A)$.

The set of arguments built upon $\mathcal{T}$ is denoted $\mathcal{A}(\mathcal{T})$.

As in [11], we consider *atomic* arguments (2) and *composite* arguments (3). Moreover, we distinguish *hypothetical* arguments (1) and *built* arguments (2/3). Notice that we add a technically essential constraint on arguments that is commonly assumed in the literature, namely that each argument is consistent. Due to the abductive nature of decision making, we define and construct arguments by reasoning backwards. Therefore, arguments are minimal, i.e. they do not include irrelevant information such as sentences not used to derive the conclusion.

Contrary to other definitions of arguments (pair of premises - conclusion, sequence of rules), our definition considers that the different premises can be challenged and can be supported by tree arguments. In this way, arguments are intelligible explanations. Triple of conclusions - premises - hypotheses are simple representations of arguments. Let us consider the previous example. Some of the arguments concluding g_7 are the following:

- $B_7^1 = \langle g_7, (D(\texttt{pisa}), \texttt{Sea}(\texttt{pisa})), (D(\texttt{pisa})\rangle$;
- $B_7^2 = \langle g_7, (D(\texttt{pisa}), \texttt{Road}(\texttt{pisa})), (D(\texttt{pisa})\rangle$;
- $A_7^1 = \langle g_7, (D(\texttt{london}), \texttt{Sea}(\texttt{london})),$
  $(D(\texttt{london}), \texttt{Sea}(\texttt{london}))\rangle$;
- $A_7^2 = \langle g_7, (D(\texttt{london}), \texttt{Road}(\texttt{london})),$
  $(D(\texttt{london}), \texttt{Road}(\texttt{london}))\rangle$.

The tree argument B_7^1 is built upon one hypothetical argument supporting $D(\texttt{pisa})$ and one trivial argument supporting $\texttt{Sea}(\texttt{pisa})$. The tree argument A_7^1 is built upon two hypothetical arguments: one supporting $D(\texttt{london})$ and one supporting $\texttt{Sea}(\texttt{london})$.

Due to their structures and their natures, arguments interact with one another.

7 Interactions amongst arguments

The interactions amongst arguments may come from the incompatibility of their sentences and from their nature (hypothetical or built). We examine in turn these different sources of interaction.

Since sentences are conflicting, arguments interact with one another. For this purpose, we define the attack relation. An argument attacks another argument if the conclusion of the first one is incompatible with one sentence of the second one.

Definition 3 (Attack relation). *Let A and B be two arguments. A* attacks *B (denoted by attacks (A, B)) iff $\mathtt{conc}(A)\ \mathcal{I}\ \mathtt{sent}(B)$.*

This attack relation, often called *undermining* attack, is indirect, i.e. directed to a "subconclusion". However, the direct attack, also called *rebuttal* attack, can also be obtained [4]. By construction, two arguments with incompatible suppositions attack each other. Since each argument is consistent, it does not attack itself. The attack relation is useful to build arguments which are consistent, i.e. homogeneous explanations.

Due to the nature of argument, they are more or less hypothetical. This is the reason why we define the size of their hypotheses.

Definition 4 (Size of hypotheses). *Let A be an arguments. The* size of hypotheses *for A, denoted $\mathtt{hypsize}(A)$, is defined such that:*

1. *if A is a hypothetical argument, then $\mathtt{hypsize}(A) = 1$;*
2. *if A is a trivial argument, then $\mathtt{hypsize}(A) = 0$*
3. *if A is a tree argument and $\mathtt{sbarg}(A) = \{A_1, \ldots, A_n\}$ is the collection of subarguments of A, then $\mathtt{hypsize}(A) = \Sigma_{A' \in \mathtt{sbarg}(A)} \mathtt{hypsize}(A')$.*

The size of hypotheses for an argument does not only count the number of hypothetical subarguments which compose the argument but also counts the number of hypothetical subarguments of these subarguments, and so on.

Since arguments have different natures (hypothetical or built) and the top rules of built arguments are more or less strong, they interact with each other. For this purpose, we define the strength relation.

Definition 5 (Strength relation). *Let A_1 be a hypothetical argument, and A_2, A_3 be two built arguments.*

1. *A_2 is stronger than A_1 (denoted $A_2\ \mathcal{P}^{\mathcal{A}}\ A_1$);*
2. *If $(\mathtt{top}(A_2)\ \mathcal{P}\ \mathtt{top}(A_3)) \wedge \neg(\mathtt{top}(A_3)\ \mathcal{P}\ \mathtt{top}(A_2))$, then $A_2\ \mathcal{P}^{\mathcal{A}}\ A_3$;*
3. *If $(\mathtt{top}(A_2) \sim \mathtt{top}(A_3)) \wedge (\mathtt{hypsize}(A_2) \leq \mathtt{hypsize}(A_3))$, then $A_2\ \mathcal{P}^{\mathcal{A}}\ A_3$;*

Since $\mathcal{P}$ is a preorder on $\mathcal{T}$, $\mathcal{P}^{\mathcal{A}}$ is a preorder on $\mathcal{A}(\mathcal{T})$. The fewer the hypothetical subarguments are, the stronger the argument is. This is the reason why build arguments are preferred to hypothetical arguments. The strength relation is useful to choose (when it is possible) between homogeneous concurrent explanations, i.e. non conflicting arguments with the same conclusions.

The two previous relations can be combined to choose (when it is possible) between non-homogeneous concurrent explanations, i.e. conflicting arguments with the same conclusions.

Definition 6 (Defeats). *Let A and B be two arguments. A* defeats *B (written defeats (A, B)) iff : i) attacks (A, B); ii) $\neg(B\ \mathcal{P}^{\mathcal{A}}\ A)$. Similarly, we say that a set S of arguments defeats an argument A if A is defeated by one argument in S.*

Let us consider our previous example.The arguments in favor of Pisa (B_7^1 and B_7^2) and the arguments in favor of London (A_7^1 and A_7^2) attack each other. B_7^1 (resp. B_7^2) is stronger than A_7^1 (resp. A_7^2), since $\texttt{hypsize}(B_7^1) \leq \texttt{hypsize}(A_7^1)$ (resp. $\texttt{hypsize}(B_7^2) \leq \texttt{hypsize}(A_7^2)$). We can deduce that B_7^1/B_7^2 defeat A_7^1/A_7^2, and A_7^1/A_7^2 do not defeat B_7^1/B_7^2. Moreover, B_7^1 (resp. A_7^1) is stronger than B_7^2 (resp. A_7^2). If we only consider these four arguments, Pisa must be selected as the best alternative to achieve g_7 and the best explanation is based upon the availability of sea transports.

In this section, we have defined the interactions amongst arguments in order to give them a status.

8 Semantics

We can consider our AF abstracting away from the logical structures of arguments. This abstract AF consists of a set of arguments associated with a binary defeat relation. It can be equipped with various semantics computed by dialectical proof procedures.

Given an AF, [4] defines the following notions of "acceptable" sets of arguments:

Definition 7 (Semantics). *An argumentation framework, which is denoted AF, is a pair* $\langle \mathcal{A}, \ \textit{defeats}\ \rangle$ *where* $\mathcal{A}$ *is the set of arguments and* $\textit{defeats} \subseteq \mathcal{A} \times \mathcal{A}$ *is the defeat relationship*[4] *for AF. For* $A \in \mathcal{A}$ *an argument and* $S \subseteq \mathcal{A}$ *a set of arguments, we say that:*

- A *is* acceptable *with respect to* S *(denoted* $A \in \mathcal{S}_{\mathcal{A}}^{S}$*) iff* $\forall B \in \mathcal{A}$ *such that* $\textit{defeats}\ (B, A)$ *then* $\exists C \in S$ *such that* $\textit{defeats}\ (C, B)$*;*
- S *is* conflict-free *iff* $\forall A, B \in S\ \neg\ \textit{defeats}\ (A, B)$*;*
- S *is* admissible *iff* S *is conflict-free and* $\forall A \in S,\ A \in \mathcal{S}_{\mathcal{A}}^{S}$*;*
- S *is* preferred *iff* S *is maximally admissible;*
- S *is* complete *iff* S *is admissible and* S *contains all arguments* A *such that* S *defeats all defeaters against* A*;*
- S *is* grounded *iff* S *is minimally complete;*
- S *is* ideal *iff* S *is admissible and it is contained in every preferred set.*

The semantics of an admissible (or preferred) set of arguments is credulous, in that it sanctions a set of arguments as acceptable if it can successfully dispute every arguments against it, without disputing itself. However, there might be several conflicting admissible sets. Various sceptical semantics have been proposed for AF, notably the grounded semantics, the ideal semantics, and the sceptically preferred semantics, whereby an argument is accepted if it is a member of all maximally admissible sets of arguments.

Since some ultimate choices amongst various admissible set of alternatives are not always possible, we consider in this paper only the credulous semantics. A decision is *suggested* to reach a goal iff it is a supposition of one argument

[4] Actually, the defeat relation is called attack in [4].

in an admissible set supporting this goal. Therefore, our AF involves some ultimate choices of the decision maker amongst various admissible set of alternatives. Let us focus on the goal g_6 in the previous example, i.e. on the following sub-theory $\mathcal{T}_{g_6} = \{R_{62}, R_{61}\}$. Since $\{A^6 = \langle g_6, (D(\texttt{london})), (D(\texttt{london}))\rangle\}$ and $\{B^6 = \langle g_6, (D(\texttt{pisa})), (D(\texttt{pisa}))\rangle\}$ are both admissible, Pisa and London must be suggested as different alternatives to reach g_6.

As seen above, B_7^1 is the only admissible argument if we only consider the goal g_7. Pisa must be suggested and justified by the availability of sea transports. If we consider now the whole problem, the only admissible argument concluding g_0 is in favour of London. This argument, denoted A_0^1, contains the premise (g_1, g_2) and makes no hypothesis.

In this section, we have given a status to the arguments.

9 Procedure

A dialectical proof procedure is required to compute the model-theoretic semantics of our argumentation framework.

The procedures proposed in [12] and [4] compute the credulous semantics. Since our practical application requires to specify the internal structure of arguments, we adopt the procedure proposed in [4] to compute admissible arguments. If the procedure succeeds, we know that the argument is contained in a preferred set.

In order to compute admissible arguments in our AF, we have translated our AF in an Assumption-based AF (ABF for short). CaSAPI[5] computes the admissible semantics in the ABF by implementing the procedure proposed in [4]. Moreover, we have developed a CaSAPI meta-interpreter to relax constraints on the goals achievements and to make hypotheses in order to compute the admissible semantics in our concrete AF. The implementation of our framework, called MARGO (Multiattribute ARGumentation framework for Opinion explanation), is written in Prolog and available in GPL (GNU General Public License) at `http://margo.sourceforge.net/`.

The main predicate for argument manipulation `admissibleArgument(+C, ?P, ?H)` succeeds when `P` are the premises and `H` are the hypotheses of an admissible argument supporting the conclusion `C`. For instance, `admissibleArgument(g1,P,H)` only returns one solution such as `P = [g4, g5]` and `H = [d(london)]`. These sub-goals can be challenged. For instance, `admissibleArgument(g4,P,H)` only returns one solution such as `P = [d(london)]` and `H = [d(london)]`.

In this section, we have shown how to compute admissible arguments in our AF and how to present an intelligible explanation to the decision maker.

10 Related works

Because of space constraints, we will focus on closely related works.

[5] http://www.doc.ic.ac.uk/ dg00/casapi.html

Recently, a number of attempts have been made to use formal models of argumentation as a basis for practical reasoning [1, 2, 5, 6, 10]. [1] (resp. [2]) is a mathematical (resp. a philosophical) general approach of defeasible argumentation for practical reasoning. The Belief-Desire-Intention (BDI) model of agency is the most famous model of practical reasoning. [6, 10] are formal models of argumentation built upon this model of agency. However, the simplifying assumptions made to implement modal logic specifications of BDI agents leads to the lack of a strong theoretical underpinning. The KGP model [5] adopts Knowledge, Goals, and Plans as the main components of an agent state. Contrary to the BDI model, there is no gap between the logical specification of KGP agents and their implementations. For this purpose, this model uses computational logic frameworks such as logic programming with priorities [8]. However, it deals only partially with priorities, as required by most practical applications, e.g. preferences between goals, likely of knowledge, and credulous of alternative actions. For this purpose, we have provided here a suitable revised representation of knowledge, goals and actions by using only the ABF [4]. Future investigations must make planning abilities available. For instance, the OSCAR system [7] interleaves epistemic reasoning and planning.

11 Conclusions

In this paper, we have presented a DSS which evaluates the alternatives actions, suggests some solutions, and provides an interactive and intelligible explanation of this choice. This computer system is based upon a concrete AF for practical reasoning. A logic language is used as a concrete data structure for holding the statements like knowledge, goals, and actions. Different quantitative priorities are attached to these items corresponding to the likelihood of the knowledge, the preferences between goals, and the expected utilities of actions. These concrete data structures consist of information providing the backbone of arguments. Due to the abductive nature of decision making, arguments are built by reasoning backwards. To be intelligible, arguments are defined as tree-like structures. Since an ultimate choice amongst various admissible set of alternatives is not always possible, we have adopted a credulous semantics. In order to provide present an intelligible explanation to the decision maker, we have implemented this semantics.

Future investigations must explore how the proposed architecture can be used to analyze argumentation-based negotiations.

12 Acknowledgements

This work is supported by the Sixth Framework IST programme of the EC, under the 035200 ARGUGRID project. We would like to thank the anonymous reviewers for their detailed comments on this paper.

References

[1] Leila Amgoud. A unifed setting for inference and decision: an argumentation-based approach. In *Proc. of the 5th Workshop on Computational Models of Natural Arguments*, 2005.

[2] Trevor Bench-Capon and Henry Prakken. Justifying actions by accruing arguments. In *Proc. of the 1st International Conference on Computational Models of Argument*, pages 247–258. IO Press, 2006.

[3] Sir Isaiah Berlin. *The Hedgehog and the Fox.* Simon & Schuster, 1953.

[4] Phan Minh Dung, Paolo Mancarella, and Francesca Toni. Computing ideal sceptical argumentation. Technical report, ARGUGRID, 2006.

[5] Antonis C. Kakas, Paolo Mancarella, Fariba Sadri, Kostas Stathis, and Francesca Toni. The kgp model of agency. In *Proc. of ECAI*, pages 33–37, 2004.

[6] Trevor Bench-Capon Katie Atkinson and Peter McBurney. Agent decision using argumentation about actions. Technical Report ULCS-05-006, University of Liverpool, 2005.

[7] J.L. Pollock. The logical foundations of goal-regression planning in autonomous agents. *Artificial Intelligence*, 106(2):267–335, 1998.

[8] Henry Prakken and Giovanni Sator. Argument-based logic programming with defeasible priorities. *Journal of Applied Non-classical Logics*, 7:25–75, 1997.

[9] Henry Prakken and Gerard Vreeswijk. *Handbook of Philosophical Logic*, volume 4, chapter Logical systems for defeasible argumentation, pages 219–318. Kluwer Academic Publishers, 2002.

[10] Iyad Rahwan and Leila Amgoud. An argumentation-based approach for practical reasoning. In *Proc. of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 347–354, Hakodate, Japan, 2006. ACM Press.

[11] Gerard Vreeswijk. Abstract argumentation systems. *Artificial Intelligence*, 90(1-2):225–279, 1997.

[12] Gerard Vreeswijk and Henry Prakken. Credulous and sceptical argument games for preferred semantics. In *Proc. of the 7th European Workshop on Logic for Artificial Intelligence*, number 1919 in Lecture Notes in AI, pages 239–253, Berlin, 2000. Springer Verlag.