The Hedgehog and the Fox*
An Argumentation-Based Decision Support System

Maxime Morge

Dipartimento di Informatica, Università di Pisa
via F. Buonarroti, 2 I-56127 Pisa, Italy
morge@di.unipi.it
http://maxime.morge.org

Abstract. In this paper, we present a decision support system which is built upon an argumentation framework for practical reasoning. A logic language is used as a concrete data structure for holding statements representing knowledge, goals, and decisions. Different priorities are attached to these items, corresponding to the probability of the knowledge, the preferences between goals, and the expected utilities of decisions. These concrete data structures consist of information providing the backbone of arguments. Due to the abductive nature of practical reasoning, arguments are built by reasoning backwards, and possibly by making suppositions over missing information. Moreover, arguments are defined as tree-like structures. In this way, our computer system, implemented in Prolog, suggests some solutions and provides an interactive and intelligible explanation of this choice.

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 [1]

* The author would like to thank Paolo Mancarella for his contribution on a previous version of this paper. This work is supported by the Sixth Framework IST programme of the EC, under the 035200 ARGUGRID project.

I. Rahwan, S. Parsons, and C. Reed (Eds.): ArgMAS 2007, LNAI 4946, pp. 114–131, 2008.

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 but must suggest a solution, i.e. take a decision.

In this paper, we present a Decision Support System (DSS) with the help of an example for selecting a business location. This system is built upon an Argumentation Framework (AF) for practical reasoning. A logic language is used as a concrete data structure for holding statements representing knowledge, goals, and decisions. Different priorities are attached to these items corresponding to the uncertainty of the knowledge about the circumstances, the preferences between goals, and the expected utilities of decisions. These concrete data structures consist of information providing the backbone of arguments. Due to the abductive nature of practical reasoning, arguments are built by reasoning backwards, and possibly by making suppositions over missing information. Moreover, arguments are defined as tree-like structures. In this way, our DSS, implemented in Prolog, suggests some solutions and provides an interactive and intelligible explanation of this choice.

Section 2 presents the principle of 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 discusses some related works. Section 11 concludes with some directions for future work.

2 Principle

Basically, decision makers are categorized as either "hedgehogs", which know one big thing, or "foxes", which know many little things [2]. 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 principle 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 decisions. 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*).

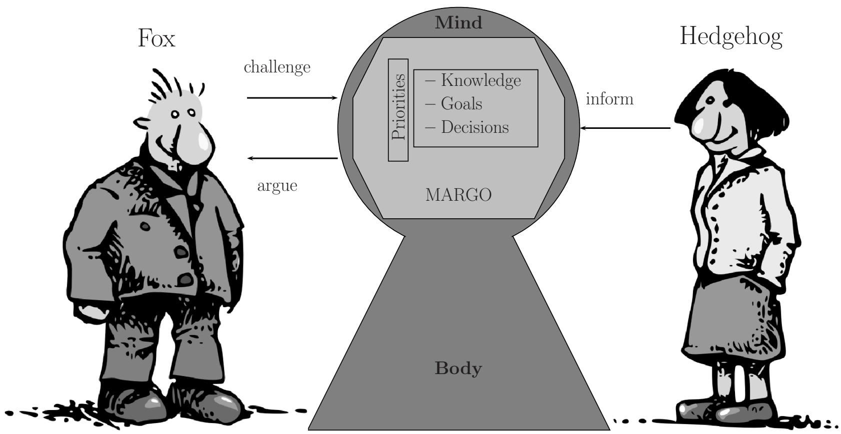

Fig. 1. Principle of the decision support system

3 Walk-Through Example

Inspired by [3], we consider here the decision making problem for selecting a suitable business location. 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 infrastructures 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 (`Location`), is addressed by a decision, i.e. a choice amongst some alternatives, i.e. Hochiminh or Hanoi (`hochiminh`, `hanoi`). The main goal (`suitable`) is split into independent sub-goals and independent sub-goals of these sub-goals. The location must offer a "good" regulation (`regulation`) and a "great" accessibility (`accessibility`). These high-level goals, which are *abstract*, reveal the user's needs. The location offers a "good" regulation, if the taxes are low (`taxes`), the permit can be easily obtained (`permit`) and an assistance is available (`assistance`). In the same way, the location offers a "good" accessibility, if the sewage is good (`sewage`) and transport are available (`transport`). 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: `Sea`(x) (the location x is accessible by sea transports), or `Road`(x) (the location x is accessible by road transports).

Figure 2 provides a simple graphical representation of the decision problem called influence diagram [4]. The elements of the decision problem, i.e. *values* (represented by rectangles with rounded corners), *decisions* (represented by rectangles) 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 where the abstract value (represented by rectangles with rounded corner and double line) aggregates the independent values in the lower level. When the structure of the decision is built, the alternatives must be identified, the preferences must be expressed and the knowledge gathered.

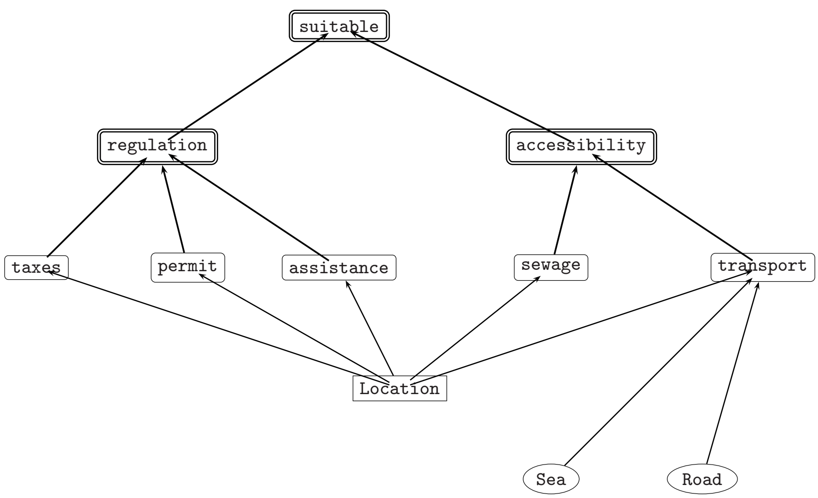

Fig. 2. Influence diagramm to structure the decision

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

4 The Object Language

Since we want to provide a computational model of argumentation for decision making and we want to instantiate it for our example, we need to specify a particular logic allowing the hedgehog to express statements representing knowledge, goals, and decisions.

The object language expresses rules and facts in logic-programming style. In order to address a decision making problem, we distinguish:

- a set of *abstract goals*, i.e. some propositional symbols which represent the abstract features that the decisions must exhibit (in the example `suitable`, `regulation`, and `accessibility`);
- a set of *concrete goals*, i.e. some propositional symbols which represent the concrete features that the decisions must exhibit (in the example `taxes`, `permit`, `assistance`, `sewage` and `transport`);
- a set of *decisions*, i.e. some predicate symbols which represent the actions which must be performed or not (in the example `Location` is the only one);
- a set of *alternatives*, i.e. some constants symbols which represent the mutually exclusive solutions for each decision (in the example `hochiminh`, or `hanoi`);
- a set of *beliefs*, i.e. some predicate symbols which represent epistemic statements of the hedgehog (in the example `Sea`, `Road`). In the language, we explicitly distinguish *assumable* beliefs (resp. *non-assumable*) beliefs, which can (resp. cannot) be taken for granted. Since the hedgehog can make the supposition that Hanoi is accessible by road, `Road`(`hanoi`) is assumable. Obviously, some beliefs are non-assumable. For instance, the hedgehog cannot make the supposition that Vienna is accessible by Sea.

Since we want to consider conflicts in this object language, we need some forms of negation. For this purpose, we consider strong negation, also called explicit or classical negation, and weak negation, also called negation as failure. A strong literal is an atomic first-order formula, possible preceded by strong negation $\neg$. A weak literal is a literal of the form $\sim L$, where L is a strong literal. $\neg L$ says "L is definitely not the case", while $\sim L$ says "There is no evidence that L is the case". In order to express in a compact way the mutual exclusion between statements, such as the different alternatives for a decision, we define the incompatibility relation (denoted by $\mathcal{I}$) as a binary relation over atomic formulas which is asymmetric. Whatever the atom L is a belief or a goal, we have $L \mathcal{I} \neg L$ and $\neg L \mathcal{I} L$, while we have $L \mathcal{I} \sim L$ but we do not have $\sim L \mathcal{I} L$. Obviously, $D_1(a_1) \mathcal{I} D_1(a_2)$ and $D_1(a_2) \mathcal{I} D_1(a_1)$, D_1 being a decision predicate, a_1 and a_2 being different[1] alternatives for D. 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 of the hedgehog about the decision making problem.

Definition 1 (Theory). *A* theory $\mathcal{T}$ *is an extended logic program, i.e a finite set of rules such as* $R : L_0 \leftarrow L_1, \ldots, L_j, \sim L_{j+1}, \ldots, \sim L_n$ *with* $n \geq 0$, *each* L_i *being a strong literal. The literal* L_0, *called the* head *of the rule, is denoted* $head(R)$. *The finite set* $\{L_1, \ldots, \sim L_n\}$, *called the* body *of the rule, is denoted*

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

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:

- *goal rules* of the form $R : G_0 \leftarrow G_1, \ldots, G_n$ with $n > 0$. Each G_i is a goal literal. The head of the rule is an abstract goal (or its strong negation). According to this rule, the abstract goal is promoted (or demoted) by the combination of goal literals in the body;
- *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 (or its strong negation). The body includes a decision literal ($D(a)$) and a possible empty set of belief literals. According to this rule, the concrete goal is promoted (or demoted) 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 hedgehog's statements, all relevant pieces of information should be taken into account, such as the likelihood of beliefs, the preferences between goals, or the expected utilities of the decisions.

In Mathematics, 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 consider that all rules are potentially defeasible and that the priorities are extra-logical and domain-specific features. The priority over concurrent rules depends of the nature of rules. Rules are *concurrent* if their heads are identical or incompatible. We define three priority relations:

- the priority over *goal rules* comes from the *preferences* overs goals. The priority of such rules corresponds to the relative importance of the combination of (sub)goals in the body as far as reaching the goal in the head is concerned;

– the priority over *epistemic rules* comes from the *uncertainty* of knowledge. The prior the rule is, the more likely the rule holds;
– the priority over *decision rules* comes from the *expected utility* of decisions. The priority of such rules corresponds to the expectation of the conditional decision in promoting/demoting the goal literal.

In order to illustrate the notions introduced previously, let us go back to the example. The goal theory, the epistemic theory, and the decision theory are represented in Table 1. A rule above another one has priority over it. To simplify the graphical representation of the theories, they are stratified in non-overlapping subsets, i.e. different levels. The *ex æquo* rules are grouped in the same level. Non-comparable rules are arbitrarily assigned to a level.

Table 1. The goal theory (upper),the epistemic theory (lower left), and the decision theory (lower right)

r_{012} : `suitable` $\leftarrow$ `regulation`, `accessibility` r_{1345} : `regulation` $\leftarrow$ `taxes`, `permit`, `assistance` r_{267} : `accessibility` $\leftarrow$ `sewage`, `transport`
r_{145} : `regulation` $\leftarrow$ `permit`, `assistance`
r_{01} : `suitable` $\leftarrow$ `regulation` r_{13} : `regulation` $\leftarrow$ `taxes` r_{26} : `accessibility` $\leftarrow$ `sewage`
r_{02} : `suitable` $\leftarrow$ `accessibility` r_{14} : `regulation` $\leftarrow$ `permit` r_{27} : `accessibility` $\leftarrow$ `transport`
r_{15} : `regulation` $\leftarrow$ `assistance`

f_1 : `Road(hochiminh)` $\leftarrow$
f_2 : `Sea(hochiminh)` $\leftarrow$ f_3 : $\neg$`Road(hochiminh)` $\leftarrow$

r_{31} : `taxes` $\leftarrow D($`hanoi`$)$ r_{42} : `permit` $\leftarrow D($`hochiminh`$)$ r_{52} : `assistance` $\leftarrow D($`hochiminh`$)$ $r_{71}(x)$: `transport` $\leftarrow D(x)$, `Sea`(x)
r_{32} : `taxes` $\leftarrow D($`hochiminh`$)$ r_{41} : `permit` $\leftarrow D($`hanoi`$)$ r_{51} : `assistance` $\leftarrow D($`hanoi`$)$ r_{61} : `sewage` $\leftarrow D($`hanoi`$)$ r_{62} : `sewage` $\leftarrow D($`hochiminh`$)$ $r_{72}(x)$: `transport` $\leftarrow D(x)$, `Road`(x)

According to the decision theory, both alternatives are relevant for the concrete goals `taxes` (r_{31} and r_{32}), `permit` (r_{41} and r_{42}), `assistance` (r_{51} and r_{52}), `sewage` (r_{61} and r_{62}), and `transport` ($r_{71}(x)$ and $r_{72}(x)$). Actually, taxes are lower in Hanoi ($r_{31}\ \mathcal{P}\ r_{32}$). The permit and the assistance are easier to obtain in Hochiminh ($r_{42}\ \mathcal{P}\ r_{41}$ and $r_{52}\ \mathcal{P}\ r_{51}$). We do not know if the sewage is better in Hochiminh or in Hanoi ($r_{61} \sim r_{62}$). Moreover, the utilities of these alternatives with respect to `transport` depends on the surrounding circumstances. Sea accessible locations have a better utility than road accessible locations ($r_{71}(x)\ \mathcal{P}\ r_{72}(x)$). Our formalism allows to capture the mutual influence of decisions over the independent goals.

According to the goal theory, achieving the goals `regulation`, and `accessibility` is required to reach `suitable` (cf. r_{012}). However, these constraints can be relaxed. The achievement of `accessibility` (resp. `regulation`) can be relaxed, $r_{012} \mathcal{P} r_{01}$ (resp. $r_{012} \mathcal{P} r_{02}$). Moreover, the achievement of `regulation` is more important than `accessibility` ($r_{01} \mathcal{P} r_{02}$). Our formalism allows to capture complex and incomplete information about the preferences amongst goals.

According to the epistemic theory, Hochiminh is accessible by sea transports (cf. f_2). Due to conflicting sources of information, the agent has conflicting beliefs about the road accessibility of Hochiminh (f_1 and f_3). The sources of information can be more or less reliable. For instance, we have $f_1 \ \mathcal{P} \ f_3$. We can notice that no information about the accessibility of Hanoi is available. Our formalism allows to capture complex (and incomplete) information about the likelihood of the surrounding circumstances. We will build now arguments upon these (incomplete) statements in order to compare the alternatives.

6 Arguments

Due to the abductive nature of the practical reasoning, we define and construct arguments by reasoning backwards, and possibly by making suppositions over missing information. Since we adopt a tree-like structure of arguments, our framework not only suggests some solutions but also provides an intelligible explanation of them for the fox.

The simplest way to define an argument is by a pair ⟨ premises, conclusion ⟩ as in [5]. 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 [6]. 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 the tree-like structure for arguments proposed in [7] and we extend it with suppositions on the missing information.

Definition 2 (Argument). *An argument is composed by a conclusion, a top rule, some premises, some suppositions, 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 assumable belief literal, then the argument built upon this ground and assumable literal is defined as follows:

   $$\mathtt{conc}(A) = L, \mathtt{top}(A) = \emptyset, \mathtt{premise}(A) = \emptyset, \mathtt{supp}(A) = \{L\}, \mathtt{sent}(A) = \{L\}.$$

   or
2. *a* built argument *built upon a 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 this rule as follows. Let $\mathtt{body}(R) = \{L_1, \ldots, L_n\}$ and $\mathtt{sbarg}(A) = \{A_1, \ldots, A_n\}$ be a collection

of arguments such that, for each $L_i \in \mathtt{body}(R)$, $\mathtt{conc}(A_i) = L_i$ *(each* A_i *is called a subargument of* A*). Then:* $\mathtt{conc}(A) = \mathtt{head}(R)$, $\mathtt{top}(A) = R$, $\mathtt{premise}(A) = \mathtt{body}(R)$, $\mathtt{supp}(A) = \cup_{A' \in \mathtt{sbarg}(A)} \mathtt{supp}(A')$, $\mathtt{sent}(A) = \cup_{A' \in \mathtt{sbarg}(A)} \mathtt{sent}(A') \cup \mathtt{body}(R) \cup \mathtt{head}(R)$.

As in [7], we consider *composite* arguments and *atomic* arguments where the top rule is a fact. Contrary to the 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 subarguments. In this way, arguments are intelligible explanations. Moreover, we distinguish *hypothetical* arguments (1) and *built* arguments (2). While the latters are built upon a top rule which is a rule (or a fact) of the theory, the formers are built upon missing information. In this way, our framework allows to reason further by making suppositions related to the unknow beliefs and over possible decisions under which arguments can be built. Due to the abductive nature of practical reasoning, we define and construct arguments by reasoning backwards. Therefore, arguments do not include irrelevant information such as sentences not used to derive the conclusion.

Let us consider the previous example. Some of the arguments concluding `transport` are depicted in Figure 3. According to the argument B_7^1 (resp. B_7^2), Hochiminh promotes the transport since this location is accessible by sea (resp. road). According to the argument A_7^1 (resp. A_7^2), Hanoi promotes the transport if we suppose that this location is accessible by sea (resp. by road). An argument can be represented as tree where the root is the conclusion (represented by a triangle) directly connected to the premises (represented by losanges) if they exist, and where leefs are either some suppositions (represented by circles) or θ[2]. Each plain arrow corresponds to a rule (or a fact) where the head node corresponds to the head of the rule and the tall nodes are in the body of the rule. While the tree argument B_7^1 (resp. B_7^2) is built upon two subarguments: one hypothetical argument supporting `Location(hochiminh)` and one trivial argument supporting `Sea(hochiminh)` (resp. `Road(hochiminh)`), the tree argument A_7^1 (resp. A_7^2) is built upon two subarguments which are hypothetical: one supporting `Location(hanoi)` and one supporting `Sea(hanoi)` (resp. `Road(hanoi)`). Neither trivial arguments nor hypothetical arguments contain subarguments. Due to their structures and their natures, arguments interact with one another.

7 Interactions between Arguments

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

Since their sentences are conflicting, arguments interact with one another. For this purpose, we define the attack relation.

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

[2] θ denotes that no literal is required.

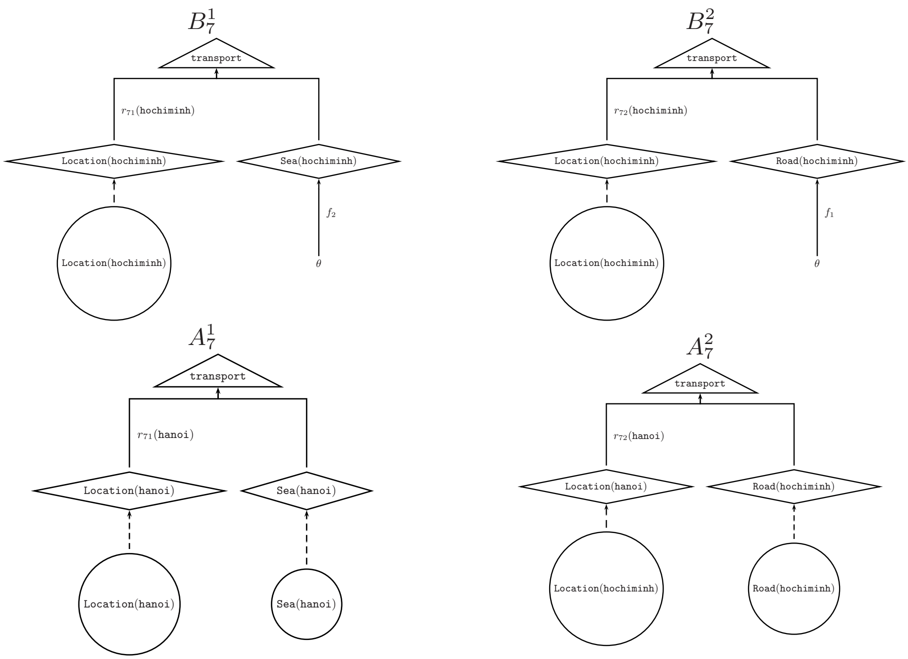

Fig. 3. Some arguments concluding `transport`

This relation encompasses both the direct (often called *rebuttal*) attack due to the incompatibility of the conclusions, and the indirect (often called *undermining*) attack, i.e. directed to a "subconclusion". According to this definition, if an argument attacks a subargument, the whole argument is attacked. The attack relation is useful to build arguments which are homogeneous explanations for the fox.

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

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

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

The size of suppositions 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 one another. 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 $(\mathtt{top}(A_2) \sim \mathtt{top}(A_3)) \wedge (\mathtt{suppsize}(A_2) < \mathtt{suppsize}(A_3))$, 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 $\mathcal{A}(\mathcal{T})$. Built arguments are preferred to hypothetical arguments. An argument is stronger than another argument if the top rule of the first argument has a proper higher priority that the top rule of the second argument or if the top rules have the same priority but the number of suppositions made in the first argument is properly smaller than the number of suppositions made in the second argument. The strength relation is useful to choose (when it is possible) between homogeneous concurrent explanations for the fox, i.e. non conflicting arguments with the same conclusions.

The two previous relations can be combined to choose (if possible) between non-homogeneous concurrent explanations for the fox, 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:*

1. *attacks (A, B);*
2. *$\neg(B \mathcal{P}^{\mathcal{A}} A)$.*

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 Hochiminh (B_7^1 and B_7^2) and the arguments in favor of Hanoi (A_7^1 and A_7^2) attack each other.

Since the top rule of B_7^1 and A_7^1 (i.e. $r_{71}(x)$) is stronger than the top rule of B_7^2 and A_7^2 (i.e. $r_{72}(x)$), B_7^1 (resp. A_7^1) defeats A_7^2 (resp. B_7^2). Moreover, B_7^1 which includes one hypothetical argument is stronger than A_7^1, which includes two hypothetical arguments. Determining whether a suggestion and an explanation are ultimately suggested to the fox requires a complete analysis of all arguments and subarguments. In this section, we have defined the interactions between 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.

Given an AF, [8] and [9] define the following notions of "acceptable" sets of arguments:

Definition 7 (Semantics). *An AF is a pair $\langle \mathcal{A}, \text{ defeats } \rangle$ where $\mathcal{A}$ is a set of arguments and defeats $\subseteq \mathcal{A} \times \mathcal{A}$ is the defeat relationship[3] for AF. For $A \in \mathcal{A}$ an argument and $S \subseteq \mathcal{A}$ a set of arguments, we say that:*

[3] Actually,the defeat relation is called attack in [8] and in [9].

– *A is* acceptable *with respect to S (denoted $A \in \mathcal{S}^S_{\mathcal{A}}$) iff $\forall B \in \mathcal{A}$, defeats (B, A) $\exists C \in S$ such that defeats (C, B);*
– *S is* conflict-free *iff $\forall A, B \in S$ $\neg$ defeats (A, B);*
– *S is* admissible *iff S is conflict-free and $\forall A \in S$, $A \in \mathcal{S}^S_{\mathcal{A}}$;*
– *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 sets.*

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 sets of alternatives are not always possible, we consider in this paper only the credulous semantics. Let us focus on the goal `sewage` in the previous example. Since the arguments supporting Hanoi and Hochiminh are admissible, both alternatives can be suggested to reach this goal. If we consider now the whole problem, the argument depicted in Figure 4 is the only one reaching `suitable` which is admissible.

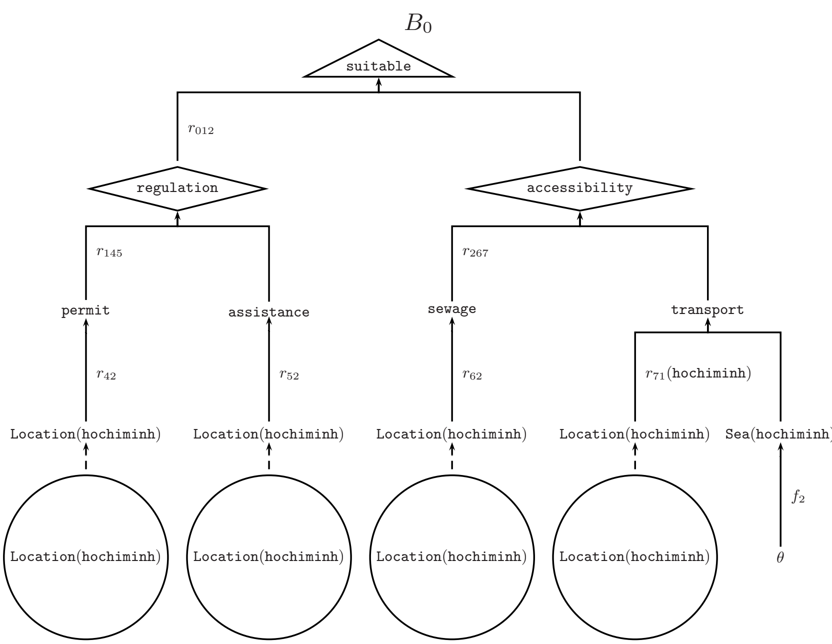

Fig. 4. An argument concluding `suitable`

In our example, there is only one admissible argument deriving the main goal. However, in the general case, a decision $D_1(a_1)$ is *suggested* iff $D_1(a_1)$ is a supposition of one argument in an admissible set deriving the main goal. Therefore, our AF involves some ultimate choices of the fox between various admissible sets of alternatives. 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 [9,10] compute the credulous semantics. Since our practical application requires to specify the internal structure of arguments, we adopt the procedure proposed in [9].

In order to compute admissible arguments in our AF, we have translated our AF in an Assumption-based AF (ABF for short). This general framework considers a deductive system $(\mathcal{L}, \mathcal{R})$ (with a language $\mathcal{L}$ and a set $\mathcal{R}$ of inference rules) augmented by a non-empty set of assumptions $\mathcal{A} \subseteq \mathcal{L}$ and a (total) mapping $\mathcal{C}on : \mathcal{A} \rightarrow \mathcal{L}$ from assumptions to their contrary. In this framework, an argument for a conclusion is a deduction of its conclusion supported by a set of assumptions. An argument attacks another argument iff the first argument supports a conclusion that is the contrary of one assumption of the second argument. The ABF corresponding to our AF is defined in the following way:

- $\mathcal{L}$ is the language described in section 4 including the names of rules and the predicate symbols `deleted` to represent when a rule does not hold;
- $\mathcal{R}$ comes from the theories and the priorities over them. If R is a goal/decision/epistemic rule then the rule r defined such as head(r) = head (R) and body(r) = body$(R) \cup \{\sim \texttt{deleted}(R)\}$ is included in $\mathcal{R}$. If R_1 and R_2 are concurrent and $R_1\ \mathcal{P}\ R_2$, then the rule r defined such as head(r) = $\texttt{deleted}(R_2)$ and body$(r) = \{\sim \texttt{deleted}(R_1)\}$ is also included in $\mathcal{R}$[4];
- $\mathcal{A}$ includes the inference rules and the the decision literals;
- $\mathcal{C}on$ comes from the incompatibility relation $\mathcal{I}$ over atomic formulas in $\mathcal{L}$.

CaSAPI[5] [12] computes the admissible semantics in the ABF by implementing the procedure originally proposed in [13]. Moreover, we have developed a CaSAPI meta-interpreter to relax the goals achievements in the priority order and to make suppositions in order to compute the admissible semantics in our concrete AF[6]. Suppose we wish to investigate whether an argument is preferred, i.e. it belongs to a preferred set. We know that it suffices to check that this argument is in an admissible set, since, by definition, a preferred set is a maximal admissible set and obviously all admissible sets are contained in a maximal admissible set. If the procedure succeeds, we know that the argument is contained in a preferred set. We can easily extend it to compute the competing semantics

[4] Our treatment of priority is inspired by [11].
[5] http://www.doc.ic.ac.uk/˜dg00/casapi.html
[6] For brevity, we do not describe this mechanism in the paper.

which have been proposed in [9]. 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/`.

In order to be computed by MARGO, the problem description must contain:

- a set of decisions, i.e. some lists which contain the alternatives courses of actions (in the example, `decisions([location(hochiminh), location(hanoi)])`);
- a set of incompatibilities, i.e. some couples which contain incompatible literals (in the example, `incompatibility(noroad(hochiminh),road(hochiminh))`);
- a set of goal rules, i.e. some triples of name - head - body which are simple Prolog representations of the goal rules in our AF (in the example, `goalrule(r012, suitable, [regulation, accesibility])`, ...);
- a set of decisions rules, i.e. some triples of name - head - body which are simple Prolog representations of the decision rules in our AF (in the example, `decisionrule(r31, taxes, [location(hanoi)])`, ...);
- a set of epistemic rules, i.e. some triples of name - head - body which are simple Prolog representations of the epistemic rules in our AF (in the example, `epistemicrule(f1,road(hochiminh),[])`, ...);
- a set of goal priorities, i.e. some ordered lists of sublists of goal rules where the rules in a previous sublists have priorities and the rules in the same sublists are *ex æquo* (in the example, `goalpriority([[r267], [r27], [r26] ])`, since $r_{267}\ \mathcal{P}\ r_{27}\ \mathcal{P}\ r_{26}, \ldots$);
- a set of decision priorities, i.e. some couples of decision rules such that the former have priority over the latter (in the example, `decisionpriority(r31,r32)`, ...);
- a set of epistemic priorities, i.e. some couples of decision rules such that the former have priority over the latter (in the example, `epistemicpriority(f1,f2)`);
- a set of possible suppositions, i.e. some couples such that the former is the name of the supposition and the latter is an assumable belief literal (in the example, `supposition(a12,road(hanoi))`, ...).

The main predicate for argument manipulation `admissibleArgument(+C, ?P, ?S)` succeeds when `P` are the premises and `S` are the suppositions of an admissible argument deriving the conclusion `C`. For instance, `admissibleArgument(suitable,P,S)` returns:

```
SUPPOSITIONS = [location(hochiminh),sea(hochiminh)],
PREMISES = [regulation,accesibility].
```

These sub-goals can be challenged. For instance, `admissibleArgument(regulation,P,S)` returns:

```
SUPPOSITIONS = [location(hochiminh)],
PREMISES = [permit,assistance].
```

The top rule of this argument is r_{145}, which is no the strongest goal rule. However, P is the strongest combination of (sub)goals which can be reach by a course of actions. In this section, we have shown how to compute admissible arguments in our AF in order to provide an interactive and intelligible explanation of the suggestion to the fox.

10 Related Works

Argumentation has been put forward as a promising approach to support decision making [14]. While influence diagrams and belief networks [15] require that all the factors relevant for a decision are identified *a priori*, arguments are defeasible or reinstantiated in the light of new information not previously available.

Contrary to the theoretical reasoning, practical reasoning is not only about whether some beliefs are true, but also about whether some actions should or should not be performed. The practical reasoning [16] follows three main steps: i) *deliberation*, i.e. the generation of goals; ii) *means-end reasoning*, i.e. the generation of plans; iii) *decision-making*, i.e. the selection of plans that will be performed to reach the selected goals. For instance, [17] proposes an AF focusing on the deliberation (closed to the principle of [18] where argumentation is implicit) and [19,20] have provided formal models for deliberation and means-end reasoning. While some frameworks are based upon defeasible logic programming (e.g. [21,22]), most of them instantiate the abstract argumentation framework of Dung [8]. Since the latter abstracts away from the internal structure of arguments in order to focus on the manner in which arguments interact, [23] instantiates an argument scheme in the context of practical reasoning in order to capture the interaction in terms of internal structure.

In this work, we have proposed an AF for decision-making. In this perspective, [24] proposes a critical survey of some computational models of argumentation over actions. For this purpose, [25,26] have considered several principles according to the different types of arguments which are considered (PROS/CONS, strong/weak, related to a positive/negative goal) are aggregated. However, contrary to our approach, the potential interaction amongst arguments, as studied in the seminal work of Dung [8] is not considered. Moreover, we allow the epistemic theory and the goal theory to be inconsistent. In this paper we have considered the example borrowed from [3] and we have adopted like [27] an abductive approach to the practical reasoning which is directly modelled within in our framework.

Finally, to the best of our knowledge, few implementation of argumentation over actions exist. CaSAPI and DeLP [7] are restricted to the theoretical reasoning. PARMENIDES[8] is a software to structure the debate over actions by adopting a particular argumentation scheme. GORGIAS [9] implements an argumentation

[7] http://lidia.cs.uns.edu.ar/DeLP
[8] http://cgi.csc.liv.ac.uk/~katie/Parmenides.html
[9] http://www.cs.ucy.ac.cy/~nkd/gorgias/

based framework to support the decision making of an agent within a modular architecture. Like the latter, MARGO incorporate abduction on missing information. Moreover, we can easily extend it to compute the competing semantics which have been proposed in [9] since we have instantiated the abstract argumentation framework of Dung.

11 Conclusions

In this paper we have presented a DSS based upon a concrete and implemented AF for practical reasoning which suggests different alternative courses of actions and provides an interactive and intelligible explanation of the choices. A logic language is used as a concrete data structure for holding statements representing knowledge, goals, and decisions. Different priorities are attached to these items corresponding to the uncertainty of the knowledge about the circumstances, the preferences between goals, and the expected utilities of decisions. These concrete data structures consist of information providing the backbone of arguments. Due to the abductive nature of practical reasoning, arguments are built by reasoning backwards, and possibly by making suppositions over missing information. To be intelligible, arguments are defined as tree-like structures. The interactions between arguments may come from the incompatibility of their sentences, from their nature (hypothetical or built) and from the priority over rules. Since an ultimate choice amongst various admissible sets of alternatives is not always possible, we have adopted a credulous semantics. In order to compute it, we have implemented our AF in Prolog.

In future works, we wants to incorporate decision-theoretic techniques within the model. Standard decision theory weighs the cost and benefits of possible outcomes with their probabilities to produce a preference on the expected utilities of the alternatives. However in many practical applications, it is not natural to give a quantitative representation of many objectives, or it could not deal with the cases of decision makers that only have partial information. Further standard decision theory provides little support in giving intelligible explanation of the choices. For this purpose, it would be best to have a hybrid approach combining both quantitative and qualitative decision theory. Argumentation provides a natural framework for these hybrid systems by providing a link between qualitative objectives and its quantitative representation.

References

1. Prakken, H., Vreeswijk, G.: Logical systems for defeasible argumentation. In: Handbook of Philosophical Logic, vol. 4, pp. 219–318. Kluwer Academic Publishers, Dordrecht (2002)
2. Berlin, S.I.: The Hedgehog and the Fox. Simon & Schuster (1953)
3. Stournaras, T. (ed.): Concrete scenarios identification & simple use cases. Deliverable document D1.1 ARGUGRID (2007)
4. Clemen, R.T.: Making Hard Decisions. Duxbury. Press (1996)

5. Amgoud, L., Cayrol, C.: A reasoning model based on the production of acceptable arguments. Annals of Maths and AI 34(1-3), 197–215 (2002)
6. Ralf Schweimeier, M.S.: Notions of attack and justified arguments for extended logic programs. In: van Harmelen, F. (ed.) Proc. of the 15th European Conference on Artificial Intelligence (ECAI 2002), Amsterdam, pp. 536–540. IOS Press, Amsterdam (2002)
7. Vreeswijk, G.: Abstract argumentation systems. Artificial Intelligence 90, 225–279 (1997)
8. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell. 77(2), 321–357 (1995)
9. Dung, P.M., Mancarella, P., Toni, F.: Computing ideal sceptical argumentation. Artificial Intelligence, Special Issue on Argumentation in Artificial Intelligence 171(10-15), 642–674 (2007)
10. Vreeswijk, G., Prakken, H.: Credulous and sceptical argument games for preferred semantics. In: Brewka, G., Moniz Pereira, L., Ojeda-Aciego, M., de Guzmán, I.P. (eds.) JELIA 2000. LNCS (LNAI), vol. 1919, pp. 239–253. Springer, Heidelberg (2000)
11. Kowalski, R., Toni, F.: Abstract argumentation. Artificial Intelligence and Law Journal Special Issue on Logical Models of Argumentation 4(3-4), 275–296 (1996)
12. Gartner, D., Toni, F.: CaSAPI: a system for credulous and sceptical argumentation. In: Simari, G., Torroni, P. (eds.) Proc. Workshop on Argumentation for Non-monotonic Reasoning, pp. 80–95 (2007)
13. Dung, P.M., Kowalski, R.A., Toni, F.: Dialectic proof procedures for assumption-based, admissible argumentation. Artificial Intelligence 170(2), 114–159 (2006)
14. Fox, J., Parsons, S.: On using arguments for reasoning about actions and values. In: Doyle, J., Thomason, R.H. (eds.) Proceedings of the Working Papers of the AAAI Spring Symposium on Qualitative Preferences in Deliberation and Practical Reasoning, Standford, pp. 55–63 (1997)
15. Oliver, R.M., Smith, J.Q. (eds.): Influence Diagrams, Belief Nets and Decision Analysis. John Wiley and Sons, Chichester (1988)
16. Raz, J. (ed.): Practical Reasoning. Oxford University Press, Oxford (1978)
17. Amgoud, L., Kaci, S.: On the generation of bipolar goals in argumentation-based negotiation. In: Rahwan, I., Moraïtis, P., Reed, C. (eds.) ArgMAS 2004. LNCS (LNAI), vol. 3366, pp. 192–207. Springer, Heidelberg (2005)
18. Thomason, R.H.: Desires and defaults: A framework for planning with inferred goals. In: Proc. of the seventh International Confenrence on Principle of Knowledge Representation and Reasoning (KR), pp. 702–713 (2000)
19. Hulstijn, J., van der Torre, L.W.N.: Combining goal generation and planning in an argumentation framework. In: Proc. of the 9h International Workshop on Non-Monotonic Reasoning (NMR 2004), pp. 212–218 (2004)
20. Rahwan, I., Amgoud, L.: An argumentation-based approach for practical reasoning. In: Proc. of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 347–354. ACM Press, New York (2006)
21. Simari, G.R., García, A.J., C., M.: Actions, planning and defeasible reasoning. In: Proc. of the 10th International Workshop on Non-Monotonic Reasoning, Whistler BC, Canada, pp. 377–384 (2004)
22. Kakas, A., Moraitis, P.: Argumentative-based decision-making for autonomous agents. In: Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), pp. 883–890. ACM Press, New York (2003)

23. Atkinson, K., Bench-Capon, T., McBurney, P.: Computational representation of practical argument. Synthese, special issue on Knowledge, Rationality and Action 152(2), 157–206 (2006)
24. Ouerdane, W., Maudet, N., Tsoukias, A.: Arguing over actions that involve multiple criteria: A critical review. In: Mellouli, K. (ed.) ECSQARU 2007. LNCS (LNAI), vol. 4724, Springer, Heidelberg (2007)
25. Amgoud, L., Prade, H.: Comparing decisions in an argumentation-based setting. In: Proc. of the 11th International Workshop on Non-Monotonic Reasoning (NMR 2006), Session on Argumentation, Dialogue, and Decision Making, Lake District, UK, pp. 426–432 (2006)
26. Amgoud, L., Prade, H.: Explaining qualitative decision under uncertainty by argumentation. In: Proc. of the 21st National Conference on Artificial Intelligence (AAAI 2006), Boston, Boston, pp. 16–20 (2006)
27. Bench-Capon, T., Prakken, H.: Justifying actions by accruing arguments. In: Proc. of the 1st International Conference on Computational Models of Argument, pp. 247–258. IOS Press, Amsterdam (2006)