Argument-based Applications to Knowledge Engineering

Daniela V. Carbogim, David Robertson, John Lee
Division of Informatics, University of Edinburgh
{danielac,dr}@dai.ed.ac.uk, john@cogsci.ed.ac.uk

Abstract

Argumentation is concerned with reasoning in the presence of imperfect information by constructing and weighing up arguments. It is an approach for inconsistency management in which conflict is explored rather than eradicated. This form of reasoning has proved applicable to many problems in knowledge engineering that involve uncertain, incomplete or inconsistent knowledge. This paper concentrates on different issues that can be tackled by automated argumentation systems and highlights important directions in argument-oriented research in knowledge engineering.

1 Introduction

One of the assumptions underlying the use of classical methods for representation and reasoning is that the information available is complete, certain and consistent. But often this is not the case. In almost every domain, there will be beliefs that are not categorical; rules that are incomplete, with unknown or implicit conditions; and conclusions that are contradictory. Therefore, we need alternative knowledge representation techniques for dealing with the problem of imperfect information.

There are two reactions to this sort of problem when designing systems. The first is to resolve conflict and restore consistency, as for instance in most research in belief revision. A second view, however, suggests that inconsistency can offer insights into rational processes and therefore should not be eradicated. Argumentation as a reasoning technique is an example of the latter. The idea behind it is that one can reason with imperfect information by constructing and weighing up arguments relevant to alternative, conflicting conclusions.

Argumentation bears a strong resemblance to certain approaches for inconsistency management, in particular to truth maintenance systems (Doyle 1979). The difference is more about a shift in emphasis than it is technical. Truth maintenance systems keep track of the reasons for deriving conclusions from a knowledge base, so they can deal with conflict by trying to explain *why* it happened. If a belief needs to be retracted (e.g. to restore consistency), truth maintenance systems can identify which are the conclusions that depend on this belief that should also be retracted. On the other hand, in argumentation it is important to make the sources of inconsistency clearer, and also to chart the course of an argument, so we can reason methodically in the face of conflict.

Formal argumentation systems are characterised by representing precisely some features of (informal) argumentation via formal languages and by applying formal inference techniques to these. Although such systems can be of different nature and have distinct aims, the notion of argument adopted by them is usually the same, corresponding to that of logical proof. In fact, the difference between argument and proof is not syntactic, but pragmatic in the sense that proofs are certain and arguments can be defeated by or preferred over others. As remarked in (Krause et al. 1995), "arguments have the *form* of logical proof, but they do not have the *force* of logical proof."

Despite the traditional interest in argumentation in many disciplines (e.g. Philosophy), computational frameworks for representing moderately complex arguments have appeared on the scene only recently. Some believe that formal argumentation has many disadvantages, because the study of formal logic can require a great deal of effort (van Eemeren et al. 1987) and its use to model real (natural language) arguments is too restrictive (Reed 1997). However, formal models of argumentation can successfully be applied as a reasoning method in certain contexts, especially if used in a lightweight manner. Lightweight uses of formality have already been advocated elsewhere

(Robertson and Agustí 1999), with the aim of applying logic to specific parts of a problem in a focused and selective way.

The goal of this paper is to illustrate the use of formal and structured semi-formal approaches to argumentation, evaluating its practical utility in knowledge engineering. Instead of taking the usual path of reviewing different proposals for solving a particular problem, here we analyse different issues that can be tackled by automated argumentation systems, briefly comparing these approaches to other paradigms found in the literature. This is not supposed to be an exhaustive survey, but an analysis of various formal representation styles that are obtained by looking at argumentation from different perspectives.

Because we take such a broad view of argumentation, the systems we describe are diverse. To guide the reader and facilitate comparison, we analyse the existing argument-based efforts in terms of the general problems they are meant to solve, which are stated at the beginning of each section.

The paper is organised as follows. In section 2, we discuss how formal argumentation can deal with non-monotonic and defeasible reasoning. Section 3 reports on the major argument-based approaches for decision making and reasoning under uncertainty. In section 4, we review some applications of argumentation in distributed settings, paying particular attention to multi-agent negotiation systems. In section 5, we focus on systems that use argumentation to support the design of an artifact, especially in the software development context. Because many argument-based systems share similar features and purposes, it is hard (if not impossible) to establish a definitive classification of which research falls into which category. However, an analysis based on our problem-oriented classification helps to highlight strengths and problems in the existing proposals. Finally, in section 6, we summarise the current state-of-the-art and speculate on important directions in argument-oriented research in knowledge engineering.

2 Applications to Non-monotonic and Defeasible Reasoning

2.a Problem Description

In this section, we consider the problem of drawing conclusions from a knowledge base in the face of incompleteness and inconsistency. Very often, the addition of new propositions into a knowledge base can invalidate previously held conclusions and introduce contradictions. In this case, reasoning is said to be non-monotonic.

Non-monotonic or defeasible reasoning[1] addresses the problem of reasoning under incompleteness and inconsistency in the sense that some conclusions can be taken back in the presence of new information. That is, a proposition can be accepted until a better reason for rejecting it is found. Approaches for dealing with non-monotonic reasoning should then have means for deciding which conclusions are justified and acceptable in a knowledge base. Here we investigate how formal argumentation models can provide this means.

2.b Defeasible Argumentation

Several approaches for formalising non-monotonic reasoning have been proposed in the literature, such as default logics (Reiter 1980; Antoniou 1998). Argumentation provides a different perspective to non-monotonic and defeasible reasoning, in which a claim is accepted or withdrawn on the basis of the arguments for and against it, and on whether these arguments can be attacked and defeated by others. This view has been characterised as *defeasible argumentation*[2] and has gained much impulse after the publication of the work of Loui (Loui 1987) and Pollock (Pollock 1987). Since then, myriad defeasible argumentation systems have been proposed (Nute 1988, 1994; Lin and Shoham 1989; Simari and Loui 1992; Brewka 1994; Dung 1995; Bondarenko et al. 1997; Freeman 1993), also motivated by research in the area of legal reasoning (Kowalski and Toni 1996, 1994;

[1]The term *defeasibility* has its origins in the context of Legal Philosophy (see (Prakken and Vreeswijk 1999, p. 10) and (Chesñevar et al. 1999, p. 3)). The ideas behind *defeasible reasoning* as it is studied in Philosophy and *non-monotonic reasoning* in Artificial Intelligence are roughly equivalent (Pollock 1987), so these terms have often been used interchangeably.

[2]A very useful view of logics for defeasible argumentation can be found in (Prakken and Vreeswijk 1999), and we will base part of this section on it. For another survey on the topic of defeasible argumentation, including a historical account of argumentation and defeasibility, see (Chesñevar et al. 1999).

Verheij 1996; Prakken 1997a,b; Prakken and Sartor 1997, 1996; Vreeswijk 1997). It is important to note that the field of Artificial Intelligence and Law has proved a fertile domain for defeasible argumentation research and applications. In this paper, we will not describe particular approaches to legal argumentation[3]; instead we concentrate on argument-based techniques for tackling defeasible reasoning.

In general, defeasible argumentation systems are intended to determine whether an argument is acceptable based on its relations to other arguments. Prakken has identified a generic conceptual framework which underlies the majority of existing defeasible argumentation systems (Prakken and Vreeswijk 1999; Prakken 1995). This framework consists of five basic notions that may not always be explicit:

- an underlying logical language;
- a concept of argument;
- a concept of conflict between arguments;
- a notion of defeat among arguments; and
- an account of the acceptability status of arguments.

The status of one argument depends on the whole set of arguments, and can be specified in two ways: *declaratively*, by defining a class of acceptable arguments; and *procedurally*, via a procedure for determining whether an argument is in this class.

A different view of procedural models was summarised by Loui in (Loui 1998), where he argues that what makes beliefs rational is not only their relations to other beliefs, but also the way in which they are built as the outcome of deliberative processes. In this sense, Loui gives an account of defeasible argumentation as resource-bounded, dialectic disputation protocols. Protocols are procedural models for constructing arguments based on notions such as which parties are involved; what are the possible moves for each party; how moves affect the outcome; how to determine if a disputation has finished; and if it has been won or lost. For the outcome to be rational, such protocols must be *fair* (e.g. parties get the same amount of resources) and *effective* (e.g. when a conclusion is established, it means that maximum resources were used in unsuccessful criticisms).

Because much work still needs to be done on the types of procedural models advocated by Loui, we base our analysis on the generic conceptual framework above. From the perspective of this framework it is possible to identify many similarities and common features between existing systems for defeasible argumentation. It is also possible to characterise the differences between these systems in terms of variations of these basic concepts. We will be looking at this framework in detail in section 2.c.

We will not be presenting the various defeasible argumentation formalisms in detail. A comprehensive account of the most relevant ones can be found in (Prakken and Vreeswijk 1999) and (Chesñevar et al. 1999). Instead, here we focus on a particular approach that is viewed as a unifying, abstract account of defeasible argumentation. The *Abstract Argumentation Framework* of Kowalski & Toni (also known as the BDTK approach) is a logic programming-based theory of argumentation that "unifies and generalises many approaches to default reasoning" (Bondarenko et al. 1997; Kowalski and Toni 1994). Most existing defeasible argumentation systems can be understood and described in terms of this framework, which will be discussed in section 2.d.

Finally, in section 2.e, we will compare the argumentation approach to other paradigms for capturing defeasible and non-monotonic reasoning found in the literature.

2.c A Conceptual Framework for Defeasible Argumentation Systems

In this section we discuss the five main concepts behind formalisms for defeasible argumentation: an underlying *logic*, notions of *argument*, *conflict* and *defeat*, and an account of the *possible status*

[3]An overview of legal applications of defeasible argumentation can be found in (Chesñevar et al. 1999, pp. 12–14). For more specific references in this area, the interested reader can refer to the Artificial Intelligence and Law Journal (`http://www.wkap.nl/journalhome.htm/0924-8463`) and to the Proceedings of the International Conference on Artificial Intelligence and Law (`http://ais.gmd.de/iaail/icail/icail.html`).

of an argument. Note that these concepts are not always explicit, and the terminology used to designate them may also vary between argumentation systems.

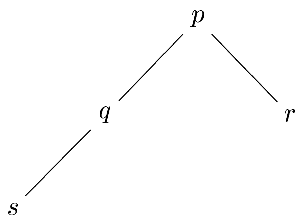
Below we present each element in some detail based on the more complete account given in (Prakken 1995; Prakken and Vreeswijk 1999).

Underlying Logic As discussed in the Introduction, formal argumentation systems are characterised by the use of formal knowledge representation and inference techniques. The underlying logic is essentially the formal logic system defining a *monotonic* consequence relation as the basis for deriving arguments. For instance, we might adopt a Horn clause resolution-based system as the underlying logic. Such systems are fundamentally deductive and therefore monotonic.

Arguments Arguments correspond to proofs in the underlying formal system. Consider, for example, the set of Horn clauses below:

$$p \leftarrow q \wedge r \qquad q \leftarrow s \qquad r \leftarrow true \qquad s \leftarrow true$$

Then the following proof of p (depicted as a tree with lower nodes supporting the conclusion above) is said to be an argument for p.



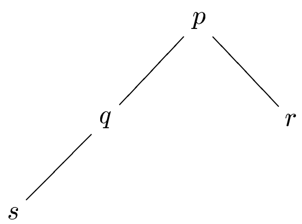
Conflict Intuitively, argumentation presupposes disagreement, which is captured in this framework by the notion of conflict. Also referred to in the literature as *attack* or *counter-argument* (Prakken and Vreeswijk 1999), *conflict* determines which conclusions in a knowledge base can be considered contradictory. For example, the sentences $married(X)$ and $bachelor(X)$ can be seen as conflicting, when instantiated by the same value for X.

It is possible to identify different types of conflict in terms of the underlying formal system, e.g. *rebuttal*. Arguments are said to be rebutting if they have contradictory conclusions. If we consider the propositions p and not_p to be conflicting, and add the following clauses to the small example above:

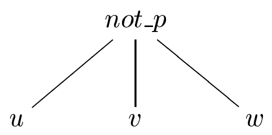
$$not\_p \leftarrow u \wedge v \wedge w \qquad u \leftarrow true \qquad v \leftarrow true \qquad w \leftarrow true$$

then the arguments A_1 and A_2 below are examples of rebutting arguments.

A_1 :



A_2 :



Defeat Because the underlying logic is monotonic, the addition of new information does not invalidate existing arguments or previously derived conclusions, so conflicting arguments may coexist in a knowledge base. In the item above, for example, we are able to derive arguments for both p and not_p. The non-monotonic character of argumentation arises from the fact that some arguments may be preferred over others, and we should have means to decide which of these arguments are acceptable.

The notion of defeat is usually based on some comparative measure for arguments and a criterion based on this measure for adjudicating between conflicting arguments. One way to do this is to assign some priority order to certain clauses in a knowledge base, and to use this order to decide between arguments. For instance, if the clause $not\_p \leftarrow u \wedge v \wedge w$ has precedence over $p \leftarrow q \wedge r$, then the argument A_2 for not_p defeats the argument A_1 for p.

It has already been argued that such criteria are usually domain specific (Konolige 1988; Prakken and Sartor 1997), but in some cases it is also possible to apply generic, domain independent standards such as the specificity principle[4] (Simari and Loui 1992).

Status The goal of a defeasible argumentation system is to determine which claims and which arguments are acceptable. The notion of acceptability can vary from formalism to formalism, but intuitively an argument that defeats a conflicting argument but is also defeated by a third one is not acceptable. Therefore, it is not enough to just look at the two conflicting arguments alone to decide upon them. Instead, it is necessary to consider all relevant arguments before making a decision.

Consider the knowledge base that extends the examples above by the addition of the following clauses.

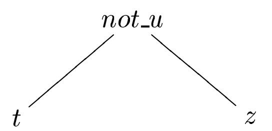
$$not\_u \leftarrow t \wedge z \quad t \leftarrow true \quad z \leftarrow true$$

Let the conflicting propositions be p and not_p; and u and not_u. Moreover, assume that the following priority ordering has been assigned to this knowledge base.

- $not\_p \leftarrow u \wedge v \wedge w$ has precedence over $p \leftarrow q \wedge r$;
- $not\_u \leftarrow t \wedge z$ has precedence over $u \leftarrow true$;
- every other clause has equal precedence.

We know from this ordering that argument A_2 for not_p defeats argument A_1 for p. However, this is not enough to decide that argument A_1 is not acceptable. This is because there might exist an argument A_3 that defeats A_2, thus restoring the validity of A_1. This is in fact the case here, as the following argument for not_u defeats A_2.

A_3 :



We can usually identify two classes of arguments with respect to their status: *acceptable* and *defeated*. An *undecided* class can sometimes be defined depending on the notion of defeat adopted (Prakken and Vreeswijk 1999). Intuitively, acceptable arguments are those arguments that can be justified with respect to all other arguments, as for instance arguments A_1 and A_3 above. In this sense, the acceptable arguments in a knowledge base can be viewed as one way of settling existing conflicts. Sometimes, e.g. in the example above, there is *exactly one* way of settling conflict according to the way preferences were defined, hence the set of acceptable arguments is unique. There may be cases, however, where conflict can be resolved in alternative ways, and therefore alternative sets of acceptable arguments may exist.

These sorts of defeasible argumentation systems can be of two types – *credulous* and *sceptical*. In a credulous system, a conclusion is considered acceptable if it follows from *some* acceptable argument. On the other hand, for a conclusion to be considered acceptable in a sceptical system, there must be an argument for it in *every* alternative set of acceptable arguments in the theory.

---

[4]The specificity principle is a priority measure in which rules that deal with specific cases are preferred over generic ones. For example, if we can derive the following conflicting arguments:

Tweety flies because Tweety is a bird

Tweety does not fly because Tweety is a penguin

then by the specificity principle the argument for *Tweety does not fly* is preferred because the fact that *Tweety is a penguin* is more specific than the fact that *Tweety is a bird*.

This conceptual sketch is in line with Dung's view that every argumentation system consists of two essential parts: an *argument generation unit (AGU)* for generating arguments; and an *argument processing unit (APU)* to decide if an argument is acceptable or not (Dung 1995). In his work, Dung argues that logic programming and non-monotonic reasoning are types of argumentation which can be formalised in an abstract way via notions of *argument* and *attack*. He proposes a method for generating meta-interpreters for argumentation systems, showing also that argumentation can be seen as logic programming. The method is simple and is described below:

- The *AGU* specifies the attack (or conflict) relationships between arguments. In (Dung 1995), these relations are considered to be primitive and represented in terms of a binary predicate *attack*: if an argument A attacks an argument B, this is expressed by $attack(A, B)$.
- The *APU* is the following logic program with negation as failure that determines whether an argument A is acceptable.

  $$acceptable(A) \leftarrow not\ defeat(A)$$
  $$defeat(A) \leftarrow attack(B, A), acceptable(B)$$

  Intuitively, an argument is acceptable if it cannot be shown to be defeated, i.e. if there is no acceptable argument that defeats it. This captures the idea that an argument A can be attacked by another argument, which in its turn may also be attacked by a third one, therefore restoring the validity of A.

From the perspective of this conceptual model we now take a closer look at the *Abstract Argumentation Framework*, a logic programming-based characterisation of defeasible argumentation which is both generic and oriented towards computation.

2.d An Abstract Account of Defeasible Argumentation

The Abstract Argumentation Framework gives a flexible way of dealing with defeasibility in argument (Bondarenko et al. 1997; Kowalski and Toni 1994, 1996). It is a language independent formalisation of defeasible argumentation that semantically characterises many approaches to default reasoning.

This framework is partly based on Dung's Argumentation Framework (Dung 1995). A fundamental difference is that in Dung's formalism the notions of argument and attack are considered as primitives, and therefore it is more abstract than Kowalski & Toni's proposal. Here we focus on the latter, which is still a unifying account of defeasible argumentation, but leaves less to be specified for someone wishing to apply it.

An Abstract Argumentation Framework is based on a monotonic deductive system $(\mathcal{L}, R)$, where $\mathcal{L}$ is a formal language and R is a set of inference rules. The following provability relation can be defined in terms of $(\mathcal{L}, R)$: $T \vdash \alpha$ if there is a *deduction* of $\alpha \in \mathcal{L}$ from a theory T via the rules in R. A theory is any set $T \subseteq \mathcal{L}$.

Definition 2.1 (Abstract Argumentation Framework) *Let $(\mathcal{L}, R)$ be a monotonic deductive system. An Abstract Argumentation Framework $(T, \mathcal{A}, ^-)$ w.r.t. $(\mathcal{L}, R)$ is an assumption-based framework defined by:*

- *a theory $T \subseteq \mathcal{L}$ representing facts or beliefs;*
- *a set of assumptions $\mathcal{A} \subseteq \mathcal{L}$, $\mathcal{A} \neq \emptyset$, that can extend any theory; and*
- *a mapping $^- : \mathcal{A} \rightarrow \mathcal{L}$ to capture the notion of the contrary of an assumption ($\overline{\alpha} \in \mathcal{L}$ represents the contrary of $\alpha \in \mathcal{A}$)*

A key motivation is that it should be possible to make explicit the assumptions on which defeasible reasoning is based. For instance, an argument which rests on such assumptions is accepted if there is no evidence to the contrary. Non-monotonicity arises when the addition of new sentences provides evidence against these assumptions, and thus arguments based on them are no longer accepted (Bondarenko et al. 1997). By analysing arguments from this perspective, "the role of argumentation is to justify the use of certain defeasible rules deriving a conclusion in preference to the use of other defeasible rules deriving conflicting conclusions" (Kowalski and Toni 1996).

Definition 2.2 (Argument) *If a conclusion $\alpha \in \mathcal{L}$ can be derived from $\Delta \subseteq \mathcal{A}$ and $T \subseteq \mathcal{L}$, then we say that $T \cup \Delta \vdash \alpha$ is an argument for α.*

Note that arguments are based on assumptions, and these assumptions can be attacked by others:

Definition 2.3 (Attack) *Let $(T, \mathcal{A}, ^-)$ be an Abstract Argumentation Framework. A set of assumptions $\Delta \subseteq \mathcal{A}$ attacks another set of assumptions $\Delta' \subseteq \mathcal{A}$ if there is $\alpha \in \Delta'$ such that $T \cup \Delta \vdash \overline{\alpha}$.*

The term *attack* used in this framework corresponds to the notion *conflict* in the conceptual sketch in section 2.c. Because an argument can only be attacked by means of its assumptions, conflicts between arguments are not symmetrical; i.e. if an argument A attacks an argument B, then B does not necessarily attack A. These types of attacks are known as *undermining (or assumption) attacks.* In this sense, all relations between arguments in the Abstract Argumentation Framework are reduced to undermining attacks, as illustrated in the following example (adapted from (Kowalski and Toni 1996) and (Robertson and Agustí 1999)).

Example 2.1 . *Consider the following theory T of an Abstract Argumentation Framework about inheritance.*

$$\begin{aligned} inherits(P, estate(B)) &\leftarrow valid\_will(W, B, P) && (1)\\ disinherited(P, estate(B)) &\leftarrow found\_guilty(P, murder(B)) && (2)\\ found\_guilty(john, murder(henry)) &\leftarrow && (3)\\ valid\_will(doc042, henry, john) &\leftarrow && (4) \end{aligned}$$

We say that a person P inherits the estate of B if there is a valid will W from B to that person. On the other hand, we say that a person P is disinherited of the estate of B if this person has been found guilty of the murder of B. In a particular inheritance case, John has been found guilty of the murder of Henry, and there exists a valid will identified as $doc042$ naming John the beneficiary of Henry's estate.

Intuitively, there is a conflict if a person P both inherits and is disinherited of some estate. It should be possible to construct two rebutting arguments for this case: one that supports the conclusion $inherits(john, estate(henry))$, and another for $disinherited(john, estate(henry))$. However, from the formal definition of attack given above, we cannot derive any conflicting argument.

Attacks are based on assumptions. Therefore, in order to allow arguments to be attacked we need to appropriately extend the expressions in the theory by adding assumptions as extra premises. Let the abducible sentences be represented by a non-provability operator of the form $cannot\_be\_shown(\alpha)$, which denotes that a sentence α is assumed to be false if it cannot be proved to be true. Note that $\overline{cannot\_be\_shown(\alpha)} = \alpha$.

Expressions 1 and 2 could then be rewritten as follows:

$$\begin{aligned} inherits(P, estate(B)) &\leftarrow valid\_will(W, B, P) \wedge \\ &\qquad cannot\_be\_shown(disinherited(P, estate(B))) && (5)\\ disinherited(P, estate(B)) &\leftarrow found\_guilty(P, murder(B)) \wedge \\ &\qquad cannot\_be\_shown(inherits(P, estate(B))) && (6) \end{aligned}$$

From definition 2.3 we now have two undermining arguments corresponding to the intuitive rebutting arguments.

There is no explicit criterion for deciding between two arguments in an Abstract Argumentation Framework. In fact, the notions of defeat and conflict coincide in the sense that every attack to an argument defeats this argument. Note that defeat can be symmetrical, so it is possible to have two arguments defeating each other. This is illustrated above, where the argument for $inherits(john, estate(henry))$ defeats the argument for $disinherited(john, estate(henry))$, and vice versa. In this sense, there are two ways of solving conflict in this inheritance base, corresponding to the following two alternative sets of acceptable arguments: one containing the argument supporting *inheritance*, and the other containing the argument supporting *disinheritance*. In a credulous system, both conclusions would be acceptable, whereas in a sceptical system neither of them would.

It should be possible, however, to represent priorities and preferences in this framework. Actually, there are two ways to prioritise an argument in terms of assumptions without altering the semantics. One way is by removing assumptions so that the argument can no longer be attacked. The second way is by introducing labels to the expressions and adding rules that talk about their priorities. A methodology for doing the latter is described in detail in (Kowalski and Toni 1996). Next we illustrate both cases.

Example 2.2 *Consider again the example 2.1. Intuitively, if John is found guilty of murdering the owner of the estate he is supposed to inherit (Henry), it is expectable that he is disinherited of that estate, even if a valid will exists. Therefore, we would like to prioritise the argument that supports disinheritance with respect to the one supporting inheritance.*

One way to do this is by removing the assumption in expression 6. Therefore, in the theory consisting of expressions 5, 2, 3 and 4 there are no arguments attacking the argument for $disinherited(john, estate(henry))$.

Another way to prioritise arguments is by talking about priorities in terms of labels. Consider the following expressions:

$$\begin{aligned} r1: inherits(P, estate(B)) &\leftarrow valid\_will(W, B, P) \wedge \\ & cannot\_be\_shown(defeated(r1(P))) \qquad (7) \\ r2: disinherited(P, estate(B)) &\leftarrow found\_guilty(P, murder(B)) \wedge \\ & cannot\_be\_shown(defeated(r2(P))) \qquad (8) \\ defeated(r1(P)) &\leftarrow cannot\_be\_shown(defeated(r2(P))) \qquad (9) \end{aligned}$$

Expression 9 intuitively corresponds to the idea of "inherits unless is disinherited of", so the argument for inheritance is defeated in case a person is proved to be disinherited of the estate under consideration. Therefore, in the theory composed of expressions 7, 8, 3, 4 and 9, the argument for $disinherited(john, estate(henry))$ *defeats the argument for* $inherits(john, estate(henry))$, *but the reverse is not the case because no clause exists for* $defeated(r2(john))$.

Having defined the notions of defeat, the arguments in an Abstract Argumentation Framework can be evaluated in terms of their ability to defend themselves against attack (Kowalski and Toni 1994). The way in which the class of *acceptable arguments* is defined can vary according to the semantics that one wants to capture. In the case of admissibility semantics, for instance, an argument is acceptable if and only if it is consistent and it attacks every argument that attacks it.

Definition 2.4 *An argument* $T \cup \Delta \vdash \alpha$ *is acceptable if and only if the set of assumptions* Δ *on which it is based is admissible.*

Definition 2.5 *A set of assumptions* $\Delta \subseteq \mathcal{A}$ *is admissible if and only if, for every* $\Delta' \subseteq \mathcal{A}$, *if* Δ' *attacks* Δ *then* Δ *attacks* $\Delta' - \Delta$.

Intuitively, in order to build an admissible argument for a conclusion α we need to construct an argument $T \cup \Delta \vdash \alpha$ and then augment the set of assumptions Δ in order to defend it against all possible attacks. Note that this is not trivial because by adding new assumptions to an argument we are also adding new potential points of attack against it.

Many other credulous and sceptical semantics for negation as failure can also be captured by adopting other definitions of acceptability[5]. In particular, different logics for default reasoning can be obtained by considering different notions of acceptability, different sets of assumptions or yet by assuming a different underlying logic.

The advantage of this framework is that it is both generic and oriented towards computation, since it can be implemented as a logic program. Recently, a parametrisable proof theory has been developed for it (Kakas and Toni 1999), where the different semantics that can be formalised via argumentation can be computed in terms of instances of these parameters.

[5]It has recently been shown in (Dimopoulos et al. 1999) that credulous reasoning under admissibility semantics is as hard as under stable semantics, but in the case of sceptical reasoning it is actually easier. Other complexity results for some of the semantics captured by the Abstract Argumentation Framework can also be found in that paper.

2.e Relation to Other Paradigms for Non-monotonic Reasoning

By appropriately instantiating the concepts described in section 2.c, argumentation frameworks can provide a characterisation of different formalisms for default reasoning, such as logic programming with negation as failure, default logic and auto-epistemic logic, among others. To reconstruct these formalisms in terms of an Abstract Argumentation Framework, we should specify the following elements (as described in definition 2.1): an underlying logic; a set of assumptions; and what it means to be the *contrary* of an assumption.

For instance, let us consider the case of default logic[6]. A default theory is based on a first-order deductive system $(\mathcal{L}', R')$ and can be defined as a pair (W, D), where W is a set of formulae in the underlying system and D is a set of default rules (Antoniou 1998). Default rules have the general form $\frac{\alpha:\beta_1,...,\beta_n}{\gamma}$, denoting that if α is true and if we can assume $\beta_1, ..., \beta_n$ to be consistent with α, then we can derive γ. Let $M\beta$ represent that *it is consistent to assume* β. A default theory (W, D) can then be described as an instance of an Abstract Argumentation Framework $(W, \mathcal{A}, ^-)$ based on a deductive system $(\mathcal{L}, R)$ as follows:

- $(\mathcal{L}, R)$ is the underlying first-order deductive system:
  - $\mathcal{L} = \mathcal{L}' \cup \{M\beta \mid \beta \in \mathcal{L}'\}$
  - $R = R' \cup \{\frac{\alpha, M\beta_1,...,M\beta_n}{\gamma} \mid \frac{\alpha:\beta_1,...,\beta_n}{\gamma} \in D\}$
- $\mathcal{A}$ is the set of assumptions defined by $\{M\beta \mid \beta \in \mathcal{L}'\}$
- The notion of the contrary of an assumption is defined as $\overline{M\beta} = \neg\beta$.

Argumentation corresponds well to non-monotonic formalisms because it is based on the idea of conflict, where arguments can be preferred or defeated. Therefore it is quite natural to use it in this context. Recently, argumentation has also been applied to the problem of belief revision (Wassermann 1999), where the framework presented in (Loui 1998) is used in a resource-bounded belief model to decide whether an incoming belief should be accepted or not.

The argumentation paradigm, however, seems to be applicable in areas other than defeasible reasoning, as summarised in (Prakken and Vreeswijk 1999, p. 9).

> "... argumentation systems have a wider scope than just reasoning with default. Firstly, argumentation systems can be applied to any form of reasoning with contradictory information, whether the contradictions have to do with rules and exceptions or not. For instance, the contradictions may arise from reasoning with several sources of information, or they may be caused by disagreement about beliefs or about moral, ethical or political claims. Moreover, it is important that several argumentation systems allow the construction and attack of arguments that are traditionally called 'ampliative', such as inductive, analogical and abductive arguments: these reasoning forms fall outside the scope of most other non-monotonic logics."

In the following sections we will explore this *wider scope* of argumentation in other contexts.

3 Applications to Decision Making under Uncertainty

3.a Problem Description

As argued in (Fox and Krause 1992), decision making is not only about quantitative option selection, but it is a complex activity that involves many other functions, such as decision structuring, communication, and representation of values, beliefs and preferences. In particular, Fox & Krause have identified the following requirements that decision support systems should satisfy: *robustness*, *flexibility*, *accountability* and *soundness*.

What makes the problem of decision making yet more complex is the fact that the available information on which decisions are based is very likely to be imperfect and uncertain. Below we describe some ways in which uncertainty can arise in a knowledge base.

[6]The interested reader should refer to (Bondarenko et al. 1997) for a more complete account of this reconstruction in terms of the Abstract Argumentation Framework with respect to the various possible semantics.

- We can have degrees of confidence associated with the information in the knowledge base, and these measures should be propagated appropriately as we reason about it.
- Uncertainty may be present in a non-deterministic fashion, where either of two (or more) alternatives can come about, but we do not know which. This type of uncertainty is usually represented in terms of disjunctions in the knowledge base.
- Moreover, uncertainty can arise when we cannot explicitly account for the many conditions that are necessary for a rule or a relation to hold. This is usually called the *qualification problem*.

In this section, we look at the problem of decision making from the more complex perspective advocated by (Fox and Krause 1992), considering cases where the information available is uncertain in one of the three senses described above.

3.b Argumentation and Decision Making

Most standard decision theories do not address all the requirements identified by Fox & Krause appropriately. On one hand, symbolic approaches such as knowledge based expert systems are usually constructed in an *ad hoc* manner, and often considered to be brittle. On the other hand, probabilistic decision theories are not sufficiently flexible nor accountable with respect to the options considered, and therefore have limited appeal to users. In fact, psychological research indicates that people do not reason *probabilistically* when faced with uncertainty[7]. Moreover, it is not always possible to obtain precise, objective statistics in certain domains (Parsons and Fox 1997).

The argumentation paradigm has been explored as an alternative approach to standard decision making theories, where decisions are made by considering arguments for and against decision options. As stated in (Fox and Krause 1992):

> "Argumentation captures a natural and familiar form of reasoning, and contributes to the robustness, flexibility and intelligibility of problem solving, while having a clear theoretical basis."

In fact, the *Logic of Argumentation* (Krause et al. 1995) is a well-established formal model for practical reasoning in which a structured argument rather than some summative measure is used for describing uncertainty. That is, the degree of confidence in a proposition is obtained by analysing the structure of the arguments relevant to it. The *Dialectical Argumentation System* (Freeman 1993) is also based on the same ideas and motivations, but it has been less extensively applied than the Logic of Argumentation. Both will be discussed in section 3.c.

Other argumentation-based decision theories look at decision making from the same perspective, but consider different representations of uncertainty. In section 3.d we will briefly look at some of these other approaches, in particular Haenni's *Assumption-based Systems* (Haenni 1998) and an extension of Dung's Argumentation Framework for modelling uncertainty (Ng et al. 1998).

3.c The Logic of Argumentation

The Logic of Argumentation (LA) is a qualitative approach to decision making, presented as an alternative to standard formalisms in order to overcome some of the limitations imposed by them. The development of LA[8] was largely based on Toulmin's work on informal argumentation (Toulmin 1958; Fox et al. 1992), particularly on his descriptive model of arguments which is summarised in Figure 1.

As argued in (Fox and Parsons 1998), certain characteristics of this structure make it suitable for practical reasoning in general and for decision making under uncertainty in particular. In contrast to strictly deductive mathematical reasoning, practical reasoning can involve imperfect information and inference relations other than deduction (Elvang-Goransson et al. 1993). In a

[7] See (Parsons and Fox 1997) for a more extensive discussion, including references to empirical evidence supporting this claim.

[8] The development of the Logic of Argumentation was in great part carried out at the Advanced Computer Laboratory of the Imperial Cancer Research Fund, motivated by applications in medical domains.

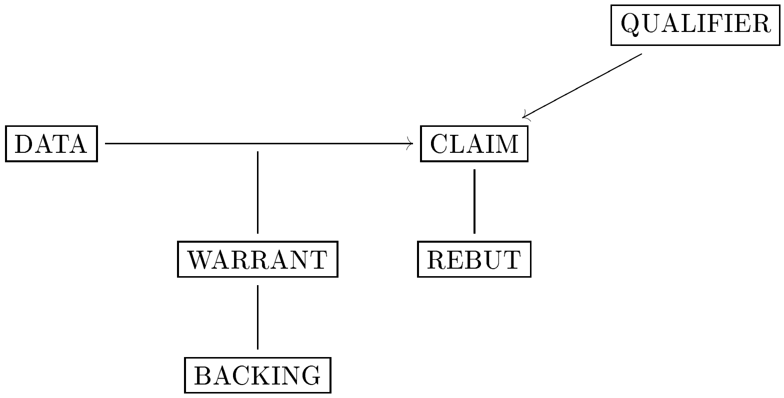


Figure 1: Toulmin's argument structure: a *claim* is supported by *data* (or evidence) and by a *warrant*, which is a general rule or principle supporting the step from data to a claim; the *backing* is a justification for the warrant, and the *rebut* is a condition where a warrant does not hold; a *qualifier* expresses the applicability of the warrant.

sense, Toulmin's model accounts for some of these issues: the idea that conclusions are followed by a qualifier suggests that degrees of confidence can be associated with claims; and contradiction can also be represented in terms of the rebut component.

3.c.1 Arguments about Beliefs

In a nutshell, the idea behind LA is to analyse the structure of the arguments that are relevant to a proposition in order to obtain a degree of confidence for it. As stated in (Krause and Clark 1993), "degrees or states of uncertainty can be viewed as a synthesis of the outcome of reasoning processes (i.e. arguments) germane to the proposition in question."

The Logic of Argumentation is based on a fragment of minimal propositional logic composed of connectives $\wedge$, $\rightarrow$ and $\neg$. In line with most formal frameworks for argumentation, an argument is defined as a proof in this logic. In a more pragmatic sense, however, an argument could be described as a "tentative proof", because it only *indicates* support for (or against) a proposition.

Each argument in LA is represented as the following structure in a Labelled Deductive System style (Gabbay 1996):

$$(St : G : S),$$

where:

- St is any formula of the underlying logic. It corresponds to the conclusion of the argument, or the *claim* in Toulmin's structure.
- G represents the grounds on which the argument is based, i.e. the proof or justification for the argument. The idea is that the sentences and formulae used to derive St in the underlying logical system are explicitly represented in G. G is therefore similar to the *data* and *warrant* supporting the *claim* in Toulmin's model.
- S is a sign, i.e. an element of a dictionary (set) of symbols or numerical values representing possible degrees of confidence in the sentence St, thus capturing the notion of *qualifier* in Toulmin's model.

A number of dictionaries of confidence measures were defined and analysed in (Fox and Parsons 1998), with emphasis on symbolic ones. An example is the so called *bounded generic* dictionary $\{+, ++\}$, in which $+$ indicates that a claim is supported, and $++$ is assigned to confirmed arguments that cannot be rebutted with respect to the grounds on which they are based. The *delta* dictionary $\{+, -\}$ is another example of a set of symbolic degrees of confidence in which the sign

$-$ represents opposing arguments, or arguments that decrease the confidence in a claim. If we consider the delta dictionary, for instance, then we can assume that the following relation holds:

$$(\neg St : G : +) \Leftrightarrow (St : G : -).$$

In summary, arguments are structures that describe how a sentence is justified. Let us assume that Δ is a knowledge base composed of argument structures. In LA, new arguments are generated at the object level of the language via an argument consequence relation denoted by $\vdash_{ACR}$. In Figure 2, we present some of the rules defining this relation in a consequent style, which can be used to generate new arguments from Δ. The interested reader can refer to (Krause et al. 1995; Fox and Parsons 1998) for a complete and detailed definition of $\vdash_{ACR}$.

(Ax) If $(St : G : S)$ is in the knowledge base, then $(St : G : S)$ is an argument itself.

$$\frac{(St{:}G{:}S) \in \Delta}{\Delta \vdash_{ACR} (St{:}G{:}S)}$$

$(\wedge\ E1)$ If we can build an argument for $St \wedge St'$ on the grounds of G and with confidence S, then we can eliminate one conjunct and build an argument for St on the same grounds G and the same degree of confidence S associated with it.

$$\frac{\Delta \vdash_{ACR} (St \wedge St'{:}G{:}S)}{\Delta \vdash_{ACR} (St{:}G{:}S)}$$

$(\rightarrow E)$ If we can build an argument for St and an argument for $St \rightarrow St'$, then we can build an argument for St'. The grounds on which St' is based are represented by the union of the grounds on which St and $St \rightarrow St'$ were derived. The degree of confidence associated with St' is obtained from a combination function with respect to the elimination of implication.

$$\frac{\Delta \vdash_{ACR} (St{:}G{:}S) \qquad \Delta \vdash_{ACR} (St \rightarrow St'{:}G'{:}S')}{\Delta \vdash_{ACR} (St'{:}G \cup G'{:}comb_{imp\ elim}(S,S'))}$$

Figure 2: The Argument Consequence Relation $\vdash_{ACR}$.

To illustrate the types of arguments that can be represented in LA, consider the following example from a medical domain (adapted from (Fox and Parsons 1998)).

Example 3.1 *Suppose that a patient has colonic polyps which could become cancerous. These beliefs can be represented in a knowledge base by the following arguments in terms of the bounded generic dictionary.*

b1:	*The patient has colonic polyps*	$(cp : \{b1\} : ++)$
b2:	*Polyps may lead to cancer*	$(cp \rightarrow ca : \{b2\} : +)$

Here cp and ca stand for "the patient has colonic polyps" and "the patient will develop cancer", respectively. The symbols b1 and b2 are labels for identifying beliefs in the knowledge base. These labels are particularly useful for representing the sentences that are used to prove or justify an argument; i.e. its grounds.

In fact, we can apply the argument consequence relation to derive, on the grounds of the arguments above, that this patient may develop cancer.

b: *The patient may develop cancer* $(ca : \{b1, b2\} : +)$

We have used the implication elimination rule in Figure 2, which corresponds to an application of Modus Ponens in which the grounds and signs have to be propagated appropriately. In this case, the sign propagation function is a minimalisation of the degree of confidence.

Thereby LA provides a way of building the arguments that are relevant to a sentence. What still needs to be defined is a mechanism for combining every distinct argument in order to obtain a single confidence measure for the sentence in question. This mechanism is also known as *aggregation* or *flattening*, and is defined in terms of flattening functions over the adopted dictionary. More specifically, if A^{St} is the set of all arguments $(St : G : Sg)$ relevant to a sentence St, then

$$Flat(A^{St}) =< St, v >,$$

where v can be an element of the given dictionary, but it can also be drawn from different ones.

The symbolic aggregation procedure defined in (Krause et al. 1995) is an example of the latter case. It combines arguments for (+) and against (−) a proposition into an element of a different dictionary (corresponding to v above), which is composed of the following linguistic terms.

$$\{certain, confirmed, probable, plausible, supported, open\}.$$

In fact, these terms closely resemble the qualifiers used by Toulmin. One advantage of this approach is that it can provide a high level summary of the available evidence without going into details of the aggregation procedure.

From the perspective of argumentation, practical reasoning in general and decision making in particular can be characterised as a two-step process in which we first construct arguments for the alternative options and then we select the most acceptable one (Elvang-Goransson et al. 1993). This idea is in line with Dung's view that every argumentation system consists of an *argument generation unit* and an *argument processing unit* (see section 2.c). The difference between this approach and the one presented in section 2 is that here we associate a "degree of acceptability" with each sentence, and therefore the argument processing step consists of picking *the most acceptable* argument instead of identifying the *acceptable* ones. It has been shown that the Logic of Argumentation can be related to other systems for non-monotonic reasoning, such as default logic. But unlike the argument-based applications to non-monotonic reasoning, LA does not in itself account for the dialectical perspective of argumentation. This aspect is now being explored in a multi-agent negotiation context, as we will see in section 4.c.

A clear mathematical semantics for argumentation and aggregation is provided in terms of category theory (Ambler 1996), so that proofs of soundness can be developed for the systems based on LA. Furthermore, other alternative semantics have also been proposed. For instance, the probabilistic semantics presented in (Parsons and Fox 1997) allows LA to represent probabilistic reasoning. In this way, the Logic of Argumentation can be used as a tool for capturing other existing formalisms, in a similar way as Kowalski & Toni's Abstract Argumentation Framework can provide argumentation-based semantics for many non-monotonic reasoning approaches.

In the context of decision support systems, the argumentation paradigm has been proved quite effective (Fox and Parsons 1998). The Logic of Argumentation has been used as the basis of an agent's internal architecture[9] and employed in a number of practical reasoning tasks, especially in medical domains where systems for supporting medical diagnosis and drug prescription are among its applications (Parsons and Fox 1997).

3.c.2 Arguments about Actions

Reasoning about beliefs – *what is the case* – is actually different from reasoning about actions – *what we ought to do* (Fox and Parsons 1998). In the first case, we can apply LA to built arguments (or tentative proofs) to support a particular conclusion. In order to reason about actions, however, we need a different notion of support, which involves values – *what is important or positive* – and expected values – *what is the expected value of doing a certain action.*

Expected values and utilities are traditional ingredients in standard decision theories. In the context of informal argumentation, these concepts were also explored in the New Rethoric (Perelman and Olbrects-Tyteca 1969), a theory that has inspired recent formal approaches such as Daphne (Grasso 1998). Daphne is a system that builds arguments to promote healthy nutrition education based on the user's values and preferences.

The following is an informal example extracted from (Fox and Parsons 1998) which extends example 3.1 and gives an argument-based characterisation of a decision making theory involving both beliefs and actions.

Example 3.2 *Suppose that a patient has colonic polyps which could become cancerous. Since cancer is life-threatening we ought to take some action to preempt this threat. Surgical excision*

[9] Fox and colleagues have developed the DOMINO model, which can be seen as an agent architecture that extends the BDI (Belief Desire Intention) model (Rao and Georgeff 1991, 1995) by incorporating procedures for decision making and plan execution based on the Logic of Argumentation (Das et al. 1996; Fox and Das 1996).

is an effective procedure for removing polyps and therefore this is an argument for carrying out surgery. Although surgery is unpleasant and has significant morbidity, this is preferable to loss of life, so surgery ought to be carried out.

Part of this reasoning is related to beliefs and could be represented in terms of LA as follows:

b1:	*The patient has colonic polyps*	$(cp : \{b1\} : ++)$
b2:	*Polyps may lead to cancer*	$(cp \rightarrow ca : \{b2\} : +)$
b3:	*Cancer may lead to loss of life*	$(ca \rightarrow ll : \{b3\} : +)$
b4:	*Surgery preempts malignancy*	$(su \rightarrow \neg(cp \rightarrow ca) : \{b4\} : ++)$
b5:	*Surgery has some side effect se*	$(su \rightarrow se : \{b5\} : ++)$

Other arguments capture the idea of value by representing whether a state is desirable or not.

v1:	*Loss of life is intolerable*	$(\neg ll : \{v1\} : ++)$
v2:	*Side effect of surgery is not desirable*	$(\neg se : \{v2\} : +)$

Arguments about the expected values of actions combine arguments about values with standard LA arguments for reasoning about beliefs.

ev1:	*Surgery should be carried out*	$(su : \{b1, b2, b3, b4, v1\} : +)$
ev2:	*Surgery should not be carried out*	$(\neg su : \{b5, v2\} : +)$

Furthermore, we should have means of expressing preferences between decision options and alternative courses of action, e.g. in terms of a special predicate 'pref'.

p1:	*Surgery side-effects is preferable to loss of life*	$(pref(se, ll) : \{v1, v2\} : ++)$
p2:	*It is preferable to carry out surgery than to not carry out surgery*	$(pref(su, \neg su) : \{ev1, ev2, p1\} : ++)$

Other types of argument can also be identified in decision making activities: closure arguments, whose grounds might include a proof that all relevant arguments have been considered; and arguments for committing to particular actions and decision options.

cl1:	*No arguments to veto surgery*	$(safe(su) : G : ++)$
co1:	*Commit to surgery*	$(do(su) : \{p2, cl1\} : ++)$

To deal with arguments about values (such as $v1$ and $v2$) and expected values (such as $ev1$ and $ev2$), Fox & Parsons have proposed a Logic of Value (LV) and a Logic of Expected Value (LEV), respectively (Fox and Parsons 1998). Arguments in LV and LEV have essentially the same format as the arguments in the Logic of Argumentation, explicitly stating the grounds on which they are based. The following schema summarises how one can reason about actions by combining belief arguments in LA with value arguments in LV in order to obtain an argument for the expected value of an action in LEV.

On the grounds of G, we can argue that action A will lead to condition C with confidence S	$(A \rightarrow C : G : S)$	(LA)
On the grounds of G', C has value V	$(C : G' : V)$	(LV)
Therefore, on the grounds of $G \cup G'$, action A has expected value E	$(A : G \cup G' : E)$	(LEV)

Apart from mechanisms for aggregating arguments about values and expected values, we also need a function that combines signs from LV and LA into a sign in LEV; i.e. with respect to the schema above, we need a function for deriving an expected value E given a value V and a confidence measure S.

Compared to the Logic of Argumentation, LV and LEV are still in a preliminary stage of development. The proposal in (Fox and Parsons 1998) concentrates on identifying which behaviour to capture rather than on providing a complete formalisation and analysis of these logics. To our knowledge, systems that involve LV and LEV have not yet been effectively implemented. The merit of this approach, however, lies in the characterisation of the different aspects of decision making in terms of argumentation. Defining such aspects via separated argumentation systems is rather intuitive and provides a more intelligible account to the problem of decision making under uncertainty.

3.c.3 LA and the Dialectical Argumentation System

Also inspired by Toulmin's argumentation model is the work of Freeman & Farley (Freeman 1993; Freeman and Farley 1992). They propose a formal theory for reasoning, making decisions, and proving and justifying claims in *weak theory domains*, i.e. domains in which knowledge is uncertain, inconsistent or incomplete. Again, the motivation for applying argumentation to deal with incomplete knowledge is that finding an adequate method for attaching numerical values to propositions, and for combining and propagating these values is a difficult task. As stated in (Freeman and Farley 1992), "argumentation can be used as a method for locating, highlighting and organizing relevant information in support of and counter to proposed claims."

In contrast to the Logic of Argumentation, an argument may be viewed not only as a structured entity, but also from a dialectical perspective. This means that an argument is not only described as a structure that organises relevant information for and against a claim, but also as a dynamic process engaged by conflicting parties as in a debate. The argument structures adopted by Freeman & Farley correspond to a slightly extended version of Toulmin's original schema (see Figure 1) and a number of qualifiers are defined in order to capture uncertainty. The extended Toulmin structures have been implemented as a dialectical argumentation system, DART, that generates arguments in a game-like, dynamic process. DART has been used to model simple legal arguments (Freeman and Farley 1996), but has not been applied to real world scenarios.

3.d Other Argumentation-based Approaches to Uncertainty

Arguing about beliefs under uncertainty is not fundamentally different from arguing about the acceptability of a claim in a non-monotonic context (as discussed in section 2). For instance, in (Ng et al. 1998) the authors present a framework for dealing with uncertain and conflicting knowledge that extends the proposals in (Dung 1995) and (Prakken and Sartor 1997).

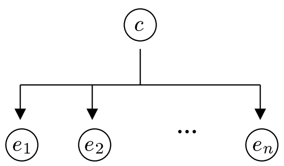
In this proposal, argumentation-based mechanisms are applied to resolve conflicts in a distributed setting, both within an agent's knowledge base and among different agents. The agents' knowledge bases are represented as extended disjunctive logic programs (Gelfond and Lifschitz 1991), where uncertainty is described by disjunctions in the head of the clauses. For instance, the clause below stands for the idea that a *a dog barks when it sees a stranger or a fire*; so if a dog barks then we know that one of these alternatives is true, but we do not know which.

$$stranger \vee fire \leftarrow dog\_barks$$

As in (Prakken and Sartor 1997), there are two types of attack: *rebut*, based on strong (classical) negation; and *undercut*, based on weak negation (or negation as failure). Defeat is based on an explicit preference hierarchy. In fact, in some specific cases (such as a single agent scenario) this framework can be proved equivalent to the one in (Prakken and Sartor 1997).

Another approach to argument-based uncertainty has recently been proposed, this time in terms of assumption-based systems. In (Haenni 1998), uncertainty is incorporated into propositional knowledge as extra assumptions, analogously to the idea in Kowalski & Toni's framework (see section 2.d) of making explicit the assumptions on which defeasible reasoning is based. This connection is not surprising, as the latter is an assumption-based system itself.

Haenni's proposal consists in transforming uncertain causal relations into clauses in an assumption-based propositional logic, and then building arguments for hypotheses based on these assumptions. For instance, the causal relations expressed by the graph below



could be represented by the following clause, which states that if cause c is true, then at least one effect among e_1, ..., e_n is also true.

$$c \rightarrow e_1 \vee e_2 \vee ... \vee e_n$$

Moreover, because some relations in a causal network can be uncertain, the effects may only come about under certain conditions, or assumptions. These assumptions are introduced as extra premises in the corresponding clauses, as shown below.

$$c \wedge a \rightarrow e_1 \vee e_2 \vee ... \vee e_n$$

An argument for an hypothesis is a set of assumptions that allow this hypothesis to be derived in the underlying propositional logic. An hypothesis is accepted or rejected based on the arguments *for* and *against* it; i.e. on the arguments that allow the hypothesis to be derived, and on the arguments that allow the falsity of the hypothesis to be derived. Note that this differs from Kowalski & Toni's approach in the sense that counter-arguments are not defined in terms of assumption attacks, but in terms of rebuttals.

Just as in the Logic of Argumentation, it is possible to aggregate the arguments relevant to an hypothesis in order to obtain a confidence measure for it. In Haenni's proposal, however, the aggregation measure is purely *quantitative*, and it can be derived by assigning prior probabilities to the assumptions and propagating them accordingly. Note that the framework also fits in the two-step process characterisation of argumentation systems discussed in the previous sections, since we first build all arguments related to an hypothesis and then, based on these arguments, we evaluate it quantitatively.

The formalism described in (Haenni 1998) has been implemented in ABEL (Assumption Based Evidential Language), a modelling language that computes symbolic and numerical arguments for an hypothesis given an expert knowledge base and a set of facts and observations (Anrig et al. 1999). ABEL has been applied mostly for reconstructing standard AI examples, in particular in the model-based diagnosis and causal modelling domains.

Argumentation can therefore be used to model decision processes under uncertainty in the sense described in section 3.a. Moreover, because the informal notion of argument is naturally connected to that of disagreement between parties, it seems that this paradigm could also be applied in distributed scenarios. This is what we explore next.

4 Applications to Multi-Agent Systems

4.a Problem Description

Intelligent software agents should be able to interact with other agents in many different ways. Such interactions usually pose a variety of issues related to information discovery, communication, reasoning, collaboration, coordination of joint approaches and social abilities. Some of these issues may be viewed as a process of achieving mutually acceptable agreements between agents (Parsons and Jennings 1997), where the nature of these agreements will vary according to the type of problem we want to address.

There are in general two types of agreement that can be attempted by agents. On one hand, agreement is about deciding on a conclusion that is acceptable to all agents involved. This sort of interaction usually takes place when there is a conflict that needs to be settled or resolved. On the other hand, agreement may be achieved in a goal-oriented type of reasoning, in which agents take a previously accepted goal as a starting point and interact in order to find an acceptable way of reaching or satisfying this goal. This sort of interaction arises when there is a common problem to be solved by the agents, who have to agree on a solution. In any case, it is important for agents to reason about their own beliefs, as well as about other agents' beliefs. So it is very likely that these interactions will be based on imperfect information in general and contradictory beliefs and intentions in particular.

Note that this way of looking at multi-agent interactions is actually in line with Walton & Krabbe's classification of dialogues. They have identified six basic types of argumentative dialogues, which can be characterised in terms of an *initial situation*, a *main goal*, and the *aims of the participants* (Walton and Krabbe 1995). One systematic way for determining the type of a dialogue is to consider whether it starts from a conflicting situation or from an open problem to be solved, in a similar way as we have characterised the types of multi-agent agreements above.

In the agent community, the problem of achieving mutually acceptable agreements between agents has often been described as *negotiation*[10]. In this context, we consider the problem of negotiation based on the two general types of agreements identified above.

4.b Argumentation-based Negotiation

Research in argumentation in multi-agent settings has been guided by the question of whether it can provide or support intelligent interaction between agents. Recently there has been much interest in applying argumentation systems to capture negotiation, since processes for reaching agreements often involve the exchange of arguments between agents.

Here we present two ways in which negotiation processes can be formalised in terms of argumentation. In section 4.c we consider protocol-based argumentation approaches, which focus on the exchange of messages between agents, and therefore are particularly useful for reaching agreements about which conclusion to accept when there is conflict. In section 4.d we consider object-based argumentation formalisms. Such formalisms focus on the construction of objects as solutions to open problems, and therefore are appropriate for reaching agreements on how to satisfy or achieve certain goals. Note that this classification is not novel, as a similar distinction on argumentation-based negotiation research was presented in (Jennings et al. 1998).

4.c Protocol-based Negotiation via Argumentation

Agent communication models or interaction protocols usually describe dialogues between agents in terms of notions that are relevant to argumentation, and therefore it is possible to look at them from an argumentation perspective. For instance, consider the case of KQML (Knowledge Query and Manipulation Language) (Finin et al. 1997; Labrou et al. 1999), an agent communication language that conveys an attitude about the content of a message when transmitting it. KQML is based on *performatives*, a concept from speech act theory in which one performs an act by uttering speech. Some KQML reserved performatives are shown in Figure 3.

Category	*Name*
Basic query	`evaluate, ask-if, ask-about, ask-one, ask-all`
Multi-response (query)	`stream-about, stream-all, eos`
Response	`reply, sorry`
Generic informational	`tell, achieve, cancel, untell, unachieve`
Generator	`standby, ready, next, rest, discard, generator`
Capability-definition	`advertise, subscribe, monitor, import, export`
Networking	`register, unregister, forward, broadcast, route`

Figure 3: Some KQML performatives classified into seven categories (from (Finin et al. 1997)).

More commonly, however, interaction protocols are only a part of argument-based negotiation models, which is used for dealing with communication issues. Negotiation formalisms normally extend single-agent argumentation frameworks (of the types presented in the previous sections) by using these to generate arguments which will be passed to other agents via some communication protocol, thus providing an argument-based approach for reasoning with imperfect information in a distributed setting.

For instance, the framework presented in (Móra et al. 1998) extends the single-agent declarative argumentation framework in (Prakken and Sartor 1997) to deal with cooperation among agents. Analogously to its single-agent counterpart, the aim is to decide which conclusions are acceptable in this distributed environment, also characterising the semantics of distributed logic programs in

[10]Negotiation is one of the six basic dialogue types identified in (Walton and Krabbe 1995), which starts with a conflict of interests and has settling (or making a deal) as its main goal. The multi-agent community adopts a broader view of negotiation, usually defined as a general process for achieving agreements. This definition subsumes other types of dialogues described by Walton & Krabbe, such as *deliberation* and *persuasion*, but is still compatible with their position: "negotiation dialogues may profit both from inquiries and from persuasion dialogues as subdialogues" (Walton and Krabbe 1995, p. 73).

terms of argumentation. In this case, however, agents can cooperate by looking for support from other agents when trying to build arguments. Agents are defined as extended logic programs, so they cooperate by asking other agents to infer certain conclusions necessary to complete a proof. The communication process is implemented via an argumentation protocol based on five speech acts: `ask`, `reply`, `propose`, `oppose` and `agree`.

The approach defined in (Schroeder 1999a) is also based on the same declarative framework described in section 2.c. The proposal is preliminary, but it goes one step further in the direction of building *effective* operational argumentation systems, as Schroeder touches on issues such as the need to define strategies for selecting the best argument in order to reduce the number of exchanged messages. Note, however, that this is still different from Loui's notion of procedural argumentation frameworks, as the outcome will not be affected by the use of different strategies. Other practical issues considered by Schroeder include the need to increase general understanding of argumentation and logic, thus undermining some of the most common critiques of the use of formal logic in modelling arguments. He addresses this need by proposing a graphical language for dynamically visualising argumentation processes (Schroeder 1999b)[11].

In the context of decision making – where it is important to resolve conflicting objectives and to coordinate cooperative actions – negotiation has been characterised in terms of a generic process for exchanging proposals, critiques, counter proposals, explanations and meta-information. This model, proposed in (Parsons and Jennings 1997), is discussed below and sketched in Figure 4.

Proposal A proposal is the basic element of negotiation, and it usually corresponds to an offer or a request.

Critique Intuitively, to critique a proposal means to reject it and maybe to attack the parts of it which are not acceptable.

Counter-proposal A counter-proposal is a type of critique where the agent not only rejects a proposal, but also presents another (preferable) one.

Explanation An explanation is a justification or an argument for a proposal, critique or counter-proposal.

Meta-information Any piece of extra information that can be used for guiding the analysis and evaluation of proposals, such as information about preferences or values.

Note that in the protocol outlined in Figure 4 there is no explicit indication of exchange of meta-information, as this type of message can be passed at any point by any agent. Arguments (explanations) may be sent together with critiques and proposals, and are represented by the formula ϕ.

This negotiation protocol is the basis of the multi-agent decision making frameworks in (Parsons et al. 1998) and (Sierra et al. 1997), which are analysed in terms of argumentation from two different but related (and maybe complementary) perspectives.

The work in (Sierra et al. 1997) was motivated by multi-agent applications in business process management domains (Jennings et al. 1996). The emphasis in this proposal is given to the social aspects of negotiation rather than to the actual generation of proposals, so the attack relations are assumed to be primitive (as in Dung's approach). The model is based on a specific common communication language which deals with elements of persuasion (Sycara 1990) – such as `threat`, `reward` and `appeal` – that agents use to try to change each other's preferences, values and beliefs. Such changes are done in a rather domain specific manner, and some investigation on notions such as values and expected utility in the sense described in section 3.c.2 might shed some light on how persuasion could be defined in more systematic terms.

While this work focus on social elements, the framework in (Parsons and Jennings 1997) and (Parsons et al. 1998) is more concerned about providing the necessary mechanisms for implementing the negotiation process in Figure 4. More specifically, it uses the Logic of Argumentation to:

[11]The interested reader can find more information about the graphical visualisation language for argumentation at `http://www.soi.city.ac.uk/homes/msch/cgi/viz/`. Moreover, a system for cooperation between agents in business process modelling is available at `http://www.soi.city.ac.uk/homes/msch/cgi/aca/aca.html`. This application was motivated by a project for developing multi-agent models in the domain of business process management (Jennings et al. 1996), which also inspired the negotiation model in (Sierra et al. 1997) discussed later in this section.

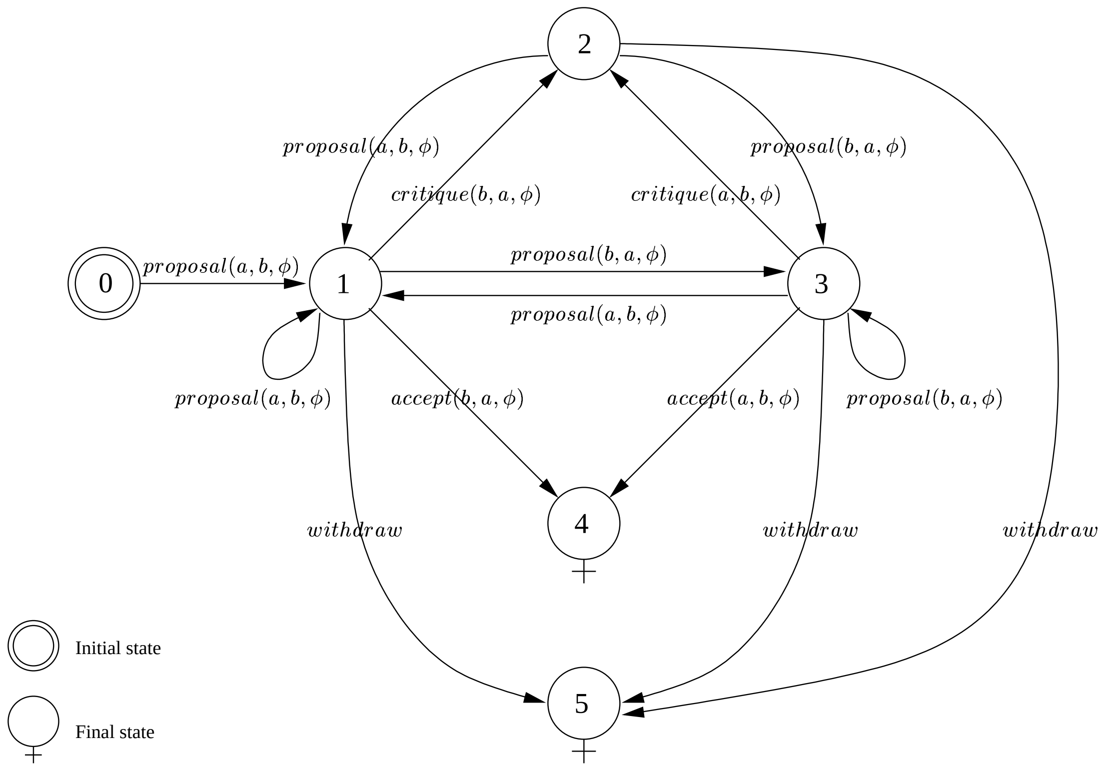


Figure 4: Negotiation protocol for two agents a and b (from (Parsons et al. 1998)).

- generate proposals, critiques, counter proposals, explanations and meta-information; and
- evaluate proposals, counter proposals and meta-information.

The Logic of Argumentation provides means of generating proposals as arguments and of evaluating them in terms of their acceptability. A crucial difference between how LA is applied here and in a single-agent environment is that now an agent has to make explicit not only the rules and facts that it used to generate an argument, but also the inference rules, because different agents might use different logics and therefore would not be able to reconstruct an argument if necessary. This issue is tackled by adopting a uniform underlying agent architecture, the multi-context architecture. An advantage of the multi-context approach is that it is generic enough to capture other architectures, such as the BDI framework (see footnote 9).

Although argumentation systems like LA give a generic architecture for a particular style of reasoning, much domain specific expertise is required to instantiate this architecture to a domain of application. Since argumentation, in automated forms, is relatively new there does not yet exist methods for guiding application of architectures to problems, and the focus has been on more abstract argumentation theory. Focusing on abstract methods rather than on problems seems to be an issue for software engineering in general (Jackson 1994; Nwana and Ndumu 1999). One way to define clear methodologies for the development of argumentation systems is to emphasise the problem and domain by identifying classes of problems in which certain evaluation principles would hold and then applying argumentation in these domains. The Logic of Argumentation provides a good example of this, where a number of different symbolic dictionaries and aggregation mechanisms were identified as suitable for medical applications, allowing different argument-based systems to be implemented in this domain (Fox and Parsons 1998).

4.d Object-based Negotiation via Argumentation

Negotiation-based models for decision making can also be seen from the perspective of the object being negotiated, rather than from a communication protocol viewpoint (Jennings et al. 1998). In general, objects are formalised as collections of issues (or variables) over which agreement can be made, and the process of negotiation consists in finding an assignment to the variables that suits every agent. However, it is also possible to consider a wider, constructive view in which the object under negotiation corresponds to an argument that has to be built by agents involved in mixed-initiative tasks. This more generic view subsumes the one where objects correspond to variables, allowing other type of negotiation processes to be characterised.

One example is the contract-based negotiation model in (Carbogim and Robertson 1999). Contracts are objects that are adjusted based on reasoned arguments by the agents involved in the agreement. In this sense, negotiation is about adjusting the terms of an agreement as opposed to the protocol-oriented view of forming an agreement. The same idea is explored in (Ferguson and Allen 1994), now in the context of mixed-initiative planning. Plans are explicitly represented as arguments that can be criticised and revised by the agents in a framework for plan construction and communication. The framework used for generating and evaluating arguments is based on previous work by Pollock (Pollock 1987) and Loui (Loui 1987). Unlike most defeasible argumentation systems, it is not used to derive defeasible conclusions from a plan, but to build a plan which is the defeasible argument itself.

In summary, the idea is to construct an argument (plan) supporting a particular conclusion (goal) such that this argument is acceptable to all agents involved. We illustrate this type of reasoning in the example below (adapted from (Ferguson and Allen 1994)):

Example 4.1 *Suppose that two agents are cooperating in order to construct a plan for transporting certain supplies (x) to a particular location. To get this done, they first need to move the supplies overland to the port and then carry them by ship. A ship (s) leaves every day between 4h00 and 6h00. If the supplies are shipped by train to the ship, they will arrive at 5h00. If they are shipped by truck, they will arrive at 3h00, but it will cost three times more than if transported by train. One possible interaction between the agents is defined below:*

- *Agent A suggests to ship the supplies by train.*
- *Agent B argues that the supplies will miss the ship if it leaves at 4h00.*
- *Agent A argues that the supplies will not miss the ship if it leaves at 6h00.*
- *Agent B then suggests to ship the supplies by truck.*
- *Agent A accepts this suggestion.*

Note that the agents could go on arguing if for some reason (such as shipping by truck is too expensive) agent A does not find the proposal acceptable.

In order to build an acceptable plan, agents make proposals, evaluate suggestions and propose alternative course of actions, in a similar way as described in the protocol-based negotiation model of Figure 4. From this perspective, the difference between the two models is very subtle. But in this case reasoning is goal-oriented (in example 4.1, the goal is to load the ship with the supplies before it leaves the dock) and pragmatically, the nature of arguments (i.e. what arguments represent) in each proposal is quite distinct.

In (Ferguson and Allen 1994) this type of reasoning is formalised by means of defeasible rules representing causal knowledge. Intuitively, these rules say that if the preconditions for an action a hold at time t, then attempting a at time t causes an event e_t to happen at the next time point. Defeasibility arises because it is hard (if not impossible) to specify all the preconditions for a rule to hold, and implicit or unknown conditions can invalidate the relation. This is also referred to as the *qualification problem*, which we have already mentioned in section 3.a. In this context, defeasible rules have the following generic form.

$$Holds(precond(a), t) \wedge Try(a, t, e_t) \rightarrow Event(e_t)$$
$$\neg Holds(precond(a), t) \wedge Try(a, t, e_t) \rightarrow \neg Event(e_t)$$

Note that the definition of an event uses the material implication (denoted here by $\supset$) instead of the defeasible implication to denote that the effects of this event will hold at the next time point.

$$Event(e_t) \supset Holds(effects(e_t), n(t))$$

Using this representation, we can formalise part of the reasoning in example 4.1.

$$AtDock(x,t) \wedge AtDock(s,t) \wedge Try(load(x,s),t,e_t) \rightarrow Load(e_t,x,s) \tag{10}$$

$$Load(e_t,x,s) \supset In(x,s,n(t)) \tag{11}$$

Moreover, it is possible to capture uncertainty in terms of disjunctions.

$$AtDock(s,t) \equiv \; t < 4h00 \;\vee\; t < 5h00 \;\vee\; t < 6h00 \tag{12}$$

The fact that $\rightarrow$ is a defeasible connective is important. Agents can build arguments for a particular goal and these arguments can be attacked because they involve elements of uncertainty and defeasibility.

What is more interesting about this approach is that it allows the representation of partial plans, which are plans that do not take all the preconditions of an action into account. To build partial plans, agents can use variants of the existing causal rules, which are obtained by considering only a subset of the preconditions specified in the original relation. As a consequence, we can define a preference criterion based on the specificity principle: in case of conflicting positions, the position supported by the more specific variant (i.e. the rule in which more preconditions are taken into account) defeats the position that is based on a less specific variant of the same rule.

Example 4.2 *To illustrate this idea, we represent the possible variants of rule 10 ordered in a lattice of specificity, where the rule at the top is the most specific one.*

$(a)\; AtDock(x,t) \wedge AtDock(s,t) \wedge Try(load(x,s),t,e_t) \rightarrow Load(e_t,x,s)$

$(b)\; AtDock(s,t) \wedge Try(load(x,s),t,e_t) \rightarrow Load(e_t,x,s)$ $\quad$ $(c)\; AtDock(x,t) \wedge Try(load(x,s),t,e_t) \rightarrow Load(e_t,x,s)$

$(d)\; Try(load(x,s),t,e_t) \rightarrow Load(e_t,x,s)$

In example 4.1, agent A presents a proposal for sending the supplies by ship based on a partial plan that disregards whether the ship is in fact at the dock at the time of loading. Such a plan can be supported by variant (c) of the original rule 10.

There are other issues involved in the type of argument described in the example which are not considered in this proposal. In particular, criteria other than specificity for evaluating arguments could be useful in this domain, especially if we want to capture the idea of values and expected values of actions. In this sense, the work on representing arguments about actions as described in 3.c.2 is relevant also to this type of application.

In the next section, we explore how this *constructive* view of argumentation can be applied in a broader context.

5 Applications to Design

5.a Problem Description

Design is the process of creating an artifact, but this general definition does not capture the complex, multifaceted nature of design activities. Moran & Carroll identify four distinct paradigms in the literature which try to portray the nature of design: design as decomposition and resynthesis; design as search in a design space; design as a process of deliberation and negotiation, in which uncertainty and disagreement is intrinsic; and design as a reflective activity (Moran and Carroll 1996). They also describe a number of issues that must be considered if we are to address the various aspects inherent to the problem of design, some of which are listed below:

- how to represent changes in the problem definition;

- how to keep track of the decisions taken and assumptions made during the design process;
- how to aid communication among different participants in the process.

These issues are relevant to design processes in a variety of domains, from architectural design to engineering design and software design. In this section, we consider how to tackle them in the context of software design.

5.b Arguing about the Software Design

By looking at design as a mixed-initiative negotiation process, we can consider the artifact (in this case, the software system) to be designed as the object of a negotiation, in the same sense discussed in section 4.d. In this way, we move from the type of multi-agent applications presented there to design support environments that possibly involve many participants.

There are two significant differences between the approaches considered in this section and the multi-agent negotiation models presented earlier in section 4. First, in the context of design, less emphasis has been given to argumentation itself than to the problem being tackled (i.e. the design of a software system involving one or more parties). This is an important point, as argued in (Moran and Carroll 1996, p. 7):

> "A lot of domain-specific knowledge is needed, and the practices of design are different in different domains (...) Useful design tools need to be domain-specific, but many of the principles behind the tools are generic."

The second difference is that in the software development context, argument systems for supporting design have been applied to fairly complex scenarios.

One way to relate the use of argumentation to software design is in terms of *viewpoints* in requirements engineering (Finkelstein et al. 1994, 1992). Though viewpoints are not explicitly characterised as arguments, they involve many ideas germane to the argument paradigm. In fact, viewpoints allow multiple perspectives to be described and integrated by dealing with inconsistencies as necessary, thus preserving these different perspectives as much as possible.

Also in the context of system requirements, a number of approaches for generating safety arguments have been presented in (Krause et al. 1997). Safety arguments are normally intended to convince people that the specified system will be safe if it is implemented appropriately. Essentially, there are three types of safety arguments (MacKenzie 1996). *Inductive* arguments support that a system is safe by testing it. *Deductive* arguments correspond to mathematical proofs that the system is correct. Finally, *constructive* arguments rely upon the process of design itself, which is argued to be a safe process that results in safe outcomes. In this section, we focus on the latter form of safety arguments.

Many problems arise when we try to represent safety arguments formally, although it has been possible to obtain effective and useful results in domain specific settings. A significant number of these problems stem not from the technicalities of the chosen argumentation system but from assumptions made about its design and deployment, since the entire safety argument cannot be made internal to the formal argumentation system and the fit to its external environment must be carefully shaped. A discussion of these issues appears in (Robertson 1999).

More commonly, argumentation is embedded in design rationale and computer-supported collaborative argumentation (CSCA)[12] systems that support the development of design activities. Design rationale is about explicitly recording the reasons why an artifact was designed in a particular way. In argumentation-based design rationale, reasons are generally represented as semi-formal arguments in terms of IBIS (Issue Based Information System) models (Conklin and Begeman 1988). In section 5.c, we will discuss another argumentation-based methodology for software design rationale.

Related to design rationale and CSCA systems, argument-based mediation systems provide support for deliberative processes involving one or more participants (users), in which the main goal is to reach a decision of some sort. Examples of mediation systems are discussion fora, where it is important to argue and negotiate about different issues, including design issues. The Zeno Argumentation Framework (Gordon and Karacapilidis 1997) is an Internet-based environment

[12] See `http://kmi.open.ac.uk/~simonb/csca/` for a resource site in CSCA.

that supports structured forms of group decision making, and it has been widely applied across different domains. Zeno is also based on Toulmin's model of argument, and can be thought of as a *formal* version of IBIS in the sense that it automatically labels and qualifies positions according to arguments and preferences (i.e. determines a degree of acceptability associated with each position). There is a focus shift between systems like Zeno and the formal approaches for decision making discussed in section 3, as in the first the emphasis is on representing arguments based on different sources and perspectives rather than on generating these arguments from some set of premises.

5.c Argumentation-based Design Rationale

In (Sigman and Liu 1999), argumentation is used to connect software system requirements to the corresponding design dialogue, providing a methodology for capturing design rationale, identifying conflicts and assessing the acceptability of design options. A generic argumentation model is used to relate the components of software design to those of design dialogue. This model allows different perspectives to be represented in terms of requirements, constraints and design features. An overview of the argumentation model is sketched in Figure 5.

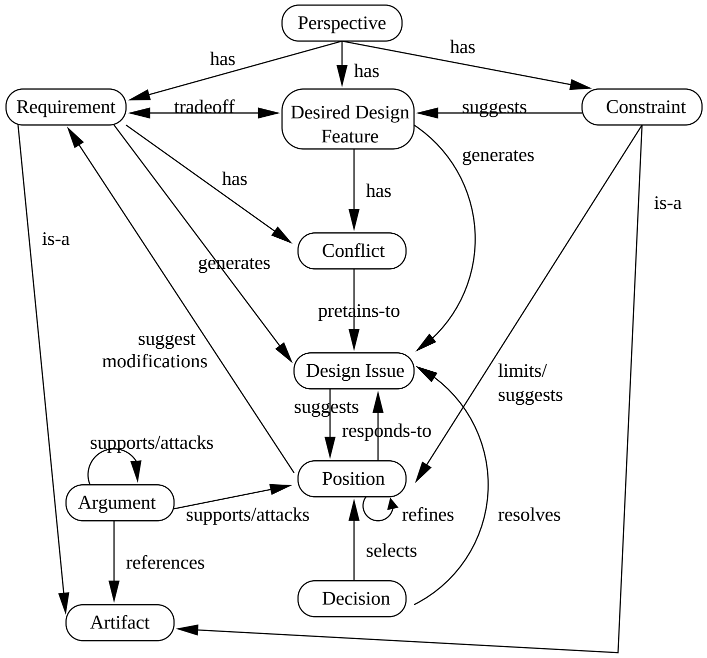
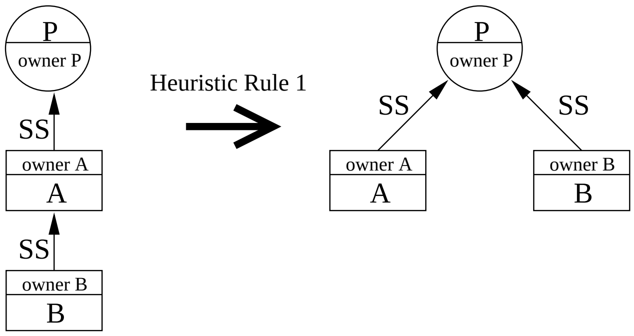


Figure 5: A software design argumentation model from (Sigman and Liu 1999).

A dialogue about a design issue can be characterised as a directed graph in which all relevant arguments for and against each alternative position are organised under the corresponding position node. These structures are referred to as *position dialogue graphs*. Positions and arguments have to explicitly state their owner, i.e. the participant that has advanced them. Moreover, linguistic labels are attached to arguments to indicate their strength. The strength measure used is that of fuzzy sets, represented in terms of the following qualitative labels: `strong attack (SA)`, `medium attack (MA)`, `inconclusive (I)`, `medium support (MS)` and `strong support (SS)`.

Some general argumentation heuristic rules provide means of reducing the position dialogue graphs in a way that all arguments are directly connected to the position node. This transformation is needed in order to identify inconsistencies as well as to assess the acceptability of the position. One example of such heuristic rules is defined below and illustrated in terms of a simplified version of position dialogue graphs.

Heuristic Rule 1 If an argument A strongly supports a position P and an argument B strongly supports argument A, then argument B strongly supports position P.



The acceptability of a position is then calculated via a *favorability factor*. The favorability factor is a function that assigns a strength measure to the position in question based on two aspects: the strength of the arguments that are relevant to this position; and the priorities previously assigned to participants, representing some idea of hierarchy among them. Comparing the favorability factors of alternative positions provides more information on which decisions can be based.

Note that this model resembles decision making approaches in many ways (see sections 3 and 4). First, the idea of using linguistic labels is similar to that proposed in the Logic of Argumentation. Secondly, the calculation of the favorability factor for a position actually corresponds to the notion of aggregation procedures in LA. Third, the idea of assigning priorities to participants is in line with the social aspects considered in the negotiation approaches in section 4.c. Finally, this proposal can be characterised as a two-step argumentation model, because all arguments relevant to a position are first gathered and then analysed in order to provide an acceptability measure for this position.

The mechanisms for manipulating arguments in this framework and in other existing approaches are essentially the same. The difference lies in the notion of argument itself and in the ways arguments are generated. In this case, an argument does not correspond to a proof, but is represented by a piece of text stating the argument. We refer to these approaches as semi-formal, since argumentation is not fully automated; or as lightweight, in the sense that formality is applied only to certain parts of the problem (e.g. automated argument evaluation).

In fact, Buckingham Shum & Hammond have argued that structured semi-formal approaches to design rationale are *useful* and *usable* (Buckingham Shum and Hammond 1994), and can play several roles in design, such as:

- structuring design problems;
- keeping track of decisions;
- facilitating communication and reasoning;
- assisting the integration of theory into design practice;
- supporting maintenance and reuse;
- exposing all assumptions (which may have been unstated) and conflicts (which may be suppressed); and
- enabling the formal incorporation of diverse types of information.

The approaches considered in this section are lightweight applications of formal argumentation which broaden the role of argumentation by carefully targeted applications of a simple formal method.

6 Conclusion

The main purpose of this paper was to analyse the practical use and usefulness of formal and structured semi formal argument-based systems in knowledge engineering. We have done this by classifying the existing efforts in terms of the problems they intend to solve, discussing whether these were actually solved or not, in which case we addressed the limitations and the remaining issues that need to be considered.

We have identified four types of problems that can be tackled by argument-based methodologies. These are:

- the problem of defeasibility in a knowledge base, where some conclusions might be withdrawn in the presence of new knowledge;
- the problem of decision making based on uncertain knowledge, where we have to decide which alternative to select;
- the problem of negotiation, where autonomous agents communicate and reason about propositions in order to reach an agreement; and
- the problem of design, where it is important to make decisions, to communicate decisions and to argue that the resulting artifact represents an acceptable solution to a particular problem.

One thing that these problems have in common is that they involve knowledge that is far from certain and complete. Potential disagreement and conflict are intrinsic to all four categories above. Therefore, the fact that conflict is the essence of argumentation might explain why the argument paradigm can be applied in these cases.

We have found many common features among the various approaches presented in this paper. Below we summarise these commonalities:

- In general, formal argumentation can be characterised as a two-step process: first, arguments are generated; then, arguments are evaluated in terms of their acceptability.
- Automated frameworks for argumentation have appeared on the scene only recently. This is probably one reason why most theories are not yet mature enough to allow applications to be developed in a systematic way. In many cases *ad hoc*, specialised solutions have been adopted in order to implement practical systems from theoretical frameworks.
- This is particularly true for argument evaluation. Generic criteria, such as the specificity principle, are not sufficient for effectively capturing the notion of argument defeat across the myriad domains in which argumentation is applicable. Therefore, many theoretical formalisms tend to leave concepts such as preferences and priorities unspecified, but without addressing the issue of how to instantiate these appropriately in order implement practical argument systems from these formalisms.
- Because argumentation is such a broad concept, many already established formalisms can be viewed from an argument perspective. Examples are KQML (section 4.c), viewpoints (section 5) and probabilistic reasoning (section 3.c.1).
- Only a few argument systems have actually been deployed in real, complex domains. Most systems have been evaluated in terms of simple benchmark problems.

There are still open research issues in each of these considerations which we believe can reflect an expected direction of development in argument-oriented research in knowledge engineering.

- The idea of argumentation as a two-step process suggests that all arguments have to be computed before they are evaluated. This may not always be the best strategy in terms of efficiency and perhaps more emphasis should be given to the constructive, resource-bounded notion of argument discussed by Loui (see section 2.b).
- In section 4.c we have discussed the need for clear methodologies for the development of argumentation systems. Note that we do not advocate a *one-size-fits-all* approach to argumentation, as we believe that the multifarious nature of argumentation cannot be captured by a uniform method. However, we would like to provide means of implementing argument theories in a systematic way, by trying to identify different methods that allow different types of argument-based systems to be developed. This may be achieved by focusing on domains and problems rather than on tasks, thus specifying domain-specific underlying theories and evaluation criteria instead of generic, domain-independent formalisms for argumentation.

- It was possible to look at certain problems in knowledge engineering from an argumentation viewpoint. This suggests that if we take a more lightweight approach to argumentation formalisms, by using them in a focused and selective way, we might broaden the scope of their applications in the field. This may be achieved by considering more flexible, semi-formal notions of arguments other than that of a proof.

- Finally, to increase the practical utility of these systems, more complex and real arguments need to be taken into account. This again might be possible to achieve by appropriately lightweight applications of argument formalisms.

The current landscape of argumentation research is scattered with tantalising glimpses of problems which may be tackled by this means, yet there are few clear guides to standard practice in this area; nor are there extensive case studies to give maps of fertile domains. Our hope is that argumentation may soon reach the level of maturity at which these become available.

Acknowledgements. The first author is supported by the Brazilian Government through the National Council for Scientific and Technological Development (CNPq), grant 200074/97-0.

The authors would like to thank the anonymous reviewers for their comments and suggestions. Thanks also to Yannis Kalfoglou for comments on an earlier draft.

References

Ambler, S. (1996). A categorical approach to the semantics of argumentation. *Mathematical Structures in Computer Science*, 6(2):167–188.

Anrig, B., Bissig, R., Haenni, R., Kohlas, J., and Lehmann, N. (1999). Probabilistig argumentation systems: introduction to assumption-based modeling with ABEL. Technical report 99-1, Institute of Informatics of the University of Fribourg. It is possible to download ABEL from the project homepage at `http://www2-iiuf.unifr.ch/tcs/ABEL/`.

Antoniou, G. (1998). A tutorial on default reasoning. *The Knowledge Engineering Review*, 13(3):225–246.

Bondarenko, A., Dung, P. M., Kowalski, R. A., and Toni, F. (1997). An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, 93(1–2):63–101.

Brewka, G. (1994). A reconstruction of Rescher's theory of formal disputation based on default logic. In Cohn, A. G., editor, *Proceedings of the 11th European Conference on Artificial Intelligence (ECAI94)*, pages 366–370.

Buckingham Shum, S. and Hammond, N. (1994). Argumentation-based design rationale: what use at what cost? *International Journal of Computer Studies*, 40(4):603–652.

Carbogim, D. and Robertson, D. (1999). Contract-based negotiation via argumentation. In *Workshop on Multi-Agent Systems in Logic Programming (MAS99) at the International Conference on Logic Programming (ICLP99)*. Also available at the workshop site `http://www.cs.sfu.ca/conf/MAS99/`.

Chesñevar, C. I., Maguitman, A. G., and Loui, R. P. (1999). Logical models of argument. *ACM Computing Surveys*, submitted. Available at `http://www.cs.wustl.edu/~loui/survey.ps`.

Conklin, J. and Begeman, M. L. (1988). gIBIS: a hypertext tool for exploratory policy discussion. *ACM Transactions on Information Systems*, 6(4):303–331.

Das, S., Fox, J., and Krause, P. (1996). A unified framework for hypothetical and practical reasoning (1): theoretical foundations. In Gabbay, D. M. and Ohlbach, H. J., editors, *Proceedings of the International Conference on Formal and Applied Practical Reasoning (FAPR96)*, pages 58–72.

Dimopoulos, Y., Nebel, B., and Toni, F. (1999). Preferred arguments are harder to compute than stable extensions. In Dean, T., editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI99)*, pages 36–41.

Doyle, J. (1979). A truth maintenance system. *Artificial Intelligence*, 12(3):232–272.

Dung, P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–358.

Elvang-Goransson, M., Krause, P., and Fox, J. (1993). Acceptability of arguments as logical uncertainty. In Clarke, M., Kruse, R., and Moral, S., editors, *Proceedings of European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU93)*, volume 747 of *Lecture Notes in Computer Science*, pages 85–90. Springer-Verlag.

Ferguson, G. and Allen, J. F. (1994). Arguing about plans: plan representation and reasoning for mixed-initiative planning. In *Proceedings of the Second International Conference on AI Planning Systems (AIPS94)*, pages 43–48, Chicago, USA.

Finin, T., Labrou, Y., and Mayfield, J. (1997). KQML as an agent communication language. In Bradshaw, J. M., editor, *Software Agents*, pages 291–316. AAAI Press/MIT Press.

Finkelstein, A., Gabbay, D., Hunter, A., Kramer, J., and Nuseibeh, B. (1994). Inconsistency handling in multi-perspective specifications. *IEEE Transactions on Software Engineering*, 20(8):569–578.

Finkelstein, A., Kramer, J., Nuseibeh, B., Finkelstein, L., and Goedicke, M. (1992). Viewpoints: a framework for integrating multiple perspectives in system development. *International Journal of Software Engineering and Knowledge Engineering*, 2(1):31–58.

Fox, J. and Das, S. (1996). A unified framework for hypothetical and practical reasoning (2): lessons from medical applications. In Gabbay, D. M. and Ohlbach, H. J., editors, *Proceedings of the International Conference on Formal and Applied Practical Reasoning (FAPR96)*, pages 73–92.

Fox, J. and Krause, P. (1992). Qualitative frameworks for decision support: lessons from medicine. *The Knowledge Engineering Review*, 7(1):19–33.

Fox, J., Krause, P., and Ambler, S. (1992). Arguments, contradictions and practical reasoning. In Neumann, B., editor, *Proceedings of the 10th European Conference on Artificial Intelligence (ECAI92)*, pages 623–627.

Fox, J. and Parsons, S. (1998). Arguing about beliefs and actions. In Hunter, A. and Parsons, S., editors, *Applications of Uncertainty Formalisms*, volume 1455 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag.

Freeman, K. (1993). *Toward formalizing dialectical argumentation*. PhD thesis, Department of Computer and Information Science of the University of Oregon, USA.

Freeman, K. and Farley, A. M. (1992). Argumentation in weak theory domains. In Ryan, K. and Sutcliffe, R., editors, *Proceedings of the 5th Irish Conference on Artificial Intelligence and Cognitive Science*. Springer-Verlag.

Freeman, K. and Farley, A. M. (1996). A model of argumentation and its application to legal reasoning. *Artificial Intelligence and Law*, 4(3–4):163–197.

Gabbay, D. M. (1996). *Labelled Deductive Systems: principles and applications. Volume 1: introduction*, volume 33 of *Oxford Logic Guides*. Oxford University Press.

Gelfond, M. and Lifschitz, V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3–4):365–385.

Gordon, T. F. and Karacapilidis (1997). The Zeno argumentation framework. In Borges, D. L. and Kaestner, C. A. A., editors, *Proceedings of the International Conference on Artificial Intelligence and Law (ICAIL-97)*, pages 10–18, University of Melbourne, Australia. More information on the Zeno System is available at the project homepage at `http://ais.gmd.de/MS/zeno/zenoSystem.html`.

Grasso, F. (1998). Exciting avocados and dull pears: combining behavioural and argumentative theory for producing effective advice. In *Proceedings of 20th Annual Meeting of the Cognitive Science Society (COG-SCI98)*, pages 436–441, Madison, USA.

Haenni, R. (1998). Modeling uncertainty with propositional assumption-based systems. In Hunter, A. and Parsons, S., editors, *Applications of Uncertainty Formalisms*, volume 1455 of *Lecture Notes in Artificial Intelligence*, pages 446–470. Springer Verlag.

Jackson, M. (1994). Problems, methods and specialisation. *Software Engineering Journal*, 9(6):249–255.

Jennings, N. R., Faratin, P., Johnson, M. J., O'Brien, P., and Wiegand, M. E. (1996). Agent-based business process management. *International Journal of Cooperative Information Systems*, 5(2–3):105–130.

Jennings, N. R., Parsons, S., Noriega, P., and Sierra, C. (1998). On argumentation-based negotiation. In *Proceedings of the International Workshop on Multi-Agents System (IWMAS)*, pages 1–7, Boston, USA.

Kakas, A. C. and Toni, F. (1999). Computing argumentation in logic programming. *Journal of Logic and Computation*, 9(4):515–562.

Konolige, K. (1988). Defeasible argumentation in reasoning about events. In Ras, Z. W. and Saitta, L., editors, *Proceedings of the Third International Symposium on Methodologies for Intelligent Systems (ISMIS 1988)*, pages 380–390, Turin, Italy.

Kowalski, R. and Toni, F. (1994). Argument and reconciliation. In *Proceedings of the Workshop of Applications of Logic Programming to Legal Reasoning at the International Symposium on Fifth Generation Computer Systems (ICOT94)*, Tokyo.

Kowalski, R. and Toni, F. (1996). Abstract argumentation. *Artificial Intelligence and Law*, 4(3–4):275–296. Special Issue on Logical Models of Argumentation, H. Prakken and G. Sartor, editors.

Krause, P., Ambler, S., Elvang-Goransson, M., and Fox, J. (1995). A logic of argumentation for reasoning under uncertainty. *Computational Intelligence*, 11(1):113–131.

Krause, P. and Clark, D. (1993). *Representing uncertain knowledge: an artificial intelligence approach*. Intellect Books, Oxford.

Krause, P., Hesketh, J., and Robertson, D. (1997). Reliable and accountable system design. *Knowledge Engineering Review*, 12(3):289–305.

Labrou, Y., Finin, T., and Peng, Y. (1999). Agent communication languages: the current landscape. *IEEE Intelligent Systems*, 14(2):45–52.

Lin, F. and Shoham, Y. (1989). Argument systems: a uniform basis for nonmonotonic reasoning. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning (KR89)*, pages 245–255. Morgan Kaufmann Publishers Inc.

Loui, R. P. (1987). Defeat among arguments: a system of defeasible inference. *Computational Intelligence*, 2:100–106.

Loui, R. P. (1998). Process and policy: resource-bounded non-demonstrative reasoning. *Computational Intelligence*, 14(1):1–38.

MacKenzie, D. (1996). A worm in the bud? Computers, systems, and the safety-case problem. Prepared for a Dibner Institute Symposium on The Spread of the Systems Approach, convened by Thomas P. Hughes. Cambridge, Mass, 3-5 May, 1996.

Móra, I. A., Alferes, J. J., and Schroeder, M. (1998). Argumentation and cooperation for distributed extended logic programs. In *Proceedings of the 7th Workshop on Non-Monotonic Reasoning (NMR98)*, Trento, Italy.

Moran, T. P. and Carroll, J. M., editors (1996). *Design rationale: concepts, techniques, and use.* Computer, Cognition, and Work. Lawrence Erlbaum Associates.

Ng, B. H.-K., Wong, K.-F., and Low, B.-T. (1998). Resolving conflicting arguments under uncertainties. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI98)*.

Nute, D. (1988). Defeasible reasoning and decision support systems. *Decision Support Systems*, 4:97–110.

Nute, D. (1994). Defeasible logic. In Gabbay, D. M., Hogger, C. J., and Robinson, J. A., editors, *Nonmonotonic Reasoning and Uncertain Reasoning*, volume 3 of *Handbook of Logic for Artificial Intelligence and Logic Programming*, pages 353–395. Oxford University Press.

Nwana, H. and Ndumu, D. (1999). A perspective on software agents research. *The Knowledge Engineering Review*, 14(2):125–142.

Parsons, S. and Fox, J. (1997). Argumentation and decision making. In *Proceedings of the IEE Coloquium on Decision Making and Problem Solving*.

Parsons, S. and Jennings, N. R. (1997). Negotiation through argumentation – a preliminary report. In *Proceedings of 2nd International Conference on Multi-Agents System*, pages 267–274, Kyoto, Japan.

Parsons, S., Sierra, C., and Jennings, N. R. (1998). Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292.

Perelman, C. and Olbrects-Tyteca, L. (1969). *The New Rethoric - a treatise on argumentation.* University of Notre Dame Press, Notre Dame/London.

Pollock, J. (1987). Defeasible reasoning. *Cognitive Science*, 11(4):481–518.

Prakken, H. (1995). From logic to dialectics in legal argument. In *Proceedings of the Fifth International Conference on Artificial Intelligence and Law*, pages 165–174, Washington DC, USA. ACM Press.

Prakken, H. (1997a). Dialectical proof theory for defeasible argumentation with defeasible priorities (preliminary report). In *Proceedings of the 4th ModelAge Workshop on Formal Models of Agents*.

Prakken, H. (1997b). *Logical tools for modelling legal argument: a study of defeasible reasoning in law.* Kluwer Academic Publishers.

Prakken, H. and Sartor, G. (1996). A dialectical model of assessing conflicting arguments in legal reasoning. *Artificial Intelligence and Law*, 4(3–4):331–368.

Prakken, H. and Sartor, G. (1997). Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-classical Logics*, 7(1):25–75. Special issue on Handling Inconsistency in Knowledge Systems.

Prakken, H. and Vreeswijk, G. (1999). Logics for defeasible argumentation. In Gabbay, D., editor, *Handbook of Philosophical Logic (to appear)*. Kluwer Academic Publishers, second edition.

Rao, A. S. and Georgeff, M. P. (1991). Modeling rational agents within a BDI-architecture. In Allen, J., Fikes, R., and Sandewall, E., editors, *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR91)*, pages 473–484. Morgan Kaufmann.

Rao, A. S. and Georgeff, M. P. (1995). Formal models and decision procedures for multi-agent systems. Technical report 61, Australian Artificial Intelligence Institute, Melbourne, Australia.

Reed, C. (1997). Representing and applying knowledge for argumentation in a social context. *AI & Society*, 11(3–4):138–154.

Reiter, R. (1980). A logic for default reasoning. *Artificial Intelligence*, 13:81–132.

Robertson, D. (1999). Can formal argumentation raise our confidence in safe design? In *Towards System Safety: Proceedings of the Seventh Safety-Critical Systems Symposium*, Huntingdon, UK.

Robertson, D. and Agustí, J. (1999). *Software blueprints: lightweight uses of logic in conceptual modelling*. Addison-Wesley (ACM Press).

Schroeder, M. (1999a). An efficient argumentation framework for negotiating autonomous agents. In *Proceedings of the Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW99).*

Schroeder, M. (1999b). Using VRML to visualise argumentation, agents and arguing agents: a preliminary report. In *Proceedings of Educational Applications of VRML (VRML99).*

Sierra, C., Jennings, N. R., Noriega, P., and Parsons, S. (1997). A framework for argumentation-based negotiation. In *Proceedings of 4th International Workshop on Agent Theories, Architectures and Languages (ATAL'97)*, pages 167–182, Rhode Island, USA.

Sigman, S. and Liu, X. F. (1999). An intelligent argumentation methodology for capturing and analysing design rationale from multiple persperctives. In *Proceedings of the 11th International Conference on Software Engineering and Knowledge Engineering (SEKE99)*, pages 227–231.

Simari, G. R. and Loui, R. P. (1992). A mathematical treatment of defeasible reasoning and its implementation. *Artificial Intelligence*, 53(2–3):125–157.

Sycara, K. (1990). Persuasive argumentation in negotiation. *Theory and Decision*, 28(3):203–242.

Toulmin, S. (1958). *The uses of argument*. Cambridge University Press, Cambridge.

van Eemeren, F. H., Grootendorst, R., and Kruiger, T. (1987). *Handbook of argumentation theory: a critical survey of classical backgrounds and modern studies.* Studies of Argumentation in Pragmatics and Discourse Analysis. Foris, Dordrecht.

Verheij, B. (1996). *Rules, reasons, arguments: formal studies of argumentation and defeat*. PhD thesis, Universiteit Maastricht.

Vreeswijk, G. (1997). Abstract argumentation systems. *Artificial Intelligence*, 90(1–2):225–279.

Walton, D. N. and Krabbe, E. C. W. (1995). *Commitment in dialogue: basic concepts of interpersonal reasoning*. State University of New York Press.

Wassermann, R. (1999). Full acceptance through argumentation – a preliminary report. In *Proceedings of the Workshop on Practical Reasoning and Rationality at the 16th International Joint Conference on Artificial Intelligence (IJCAI99).*