Argument and Reconciliation

Robert A. Kowalski, Francesca Toni
Department of Computing
Imperial College of Science, Technology and Medicine
180 Queen's Gate, London SW7 2BZ, UK
{rak,ft}@doc.ic.ac.uk

Abstract

We outline a formal theory of argumentation-theoretic reasoning which unifies and generalises many existing approaches to default reasoning, and which promises to be useful for practical reasoning in general and legal reasoning in particular. We discuss an extension of the argumentation theory to the problem of reconciling conflicting arguments.

1 Introduction

The purpose of this paper is to outline a formal theory of argumentation-theoretic reasoning, which promises to have relevance for practical reasoning in general and for legal reasoning in particular. The theory originates from research on the semantics of logic programming [9]; but its main accomplishment until now has been to provide a unifying framework for many previous approaches to the formalisation of default reasoning [4, 1, 2].

The theory is based on the conventional notion of a formal deductive system, but focuses on the problem of determining what "assumptions" can acceptably be used to extend a given set of "facts" formulated within the language of the deductive system. Some of the more noteworthy characteristics of the theory are that

- A given set of facts may have several, alternative, mutually incompatible extensions.
- An acceptable extension need not take a stand on every issue.
- For an extension to be acceptable, it is not sufficient for the set simply to be consistent with the facts. It must be able to "defend" itself against attack from other extensions.

Each of these characteristics of the theory distinguishes it from some other theories of default reasoning and argumentation. In particular, most previous approaches to the formalisation of default reasoning (including stable models [7] for logic programs, extensions [14] for default logic, stable expansions [11] for autoepistemic logic and fixed points [10] for non-monotonic modal logic) can be understood in terms of the theory as postulating that

- An extension is acceptable if it does not attack itself and attacks every assumption not in the extension.

2 Possible implications for legal theory

Classical, mathematical logic is concerned with reasoning about truths that hold universally, without exception and for all time. Attempts to apply such logic to the formalisation of human reasoning in Artificial Intelligence have long been the subject of discussion and criticism. Largely in response to these criticisms, a number of so-called "non-monotonic" logics have been developed to capture the default nature of human reasoning. It is from an attempt to capture the underlying similarities between these different non-monotonic logics that the argumentation theory we present in this paper arises.

Similar conflicts over the relevance of logic have arisen in jurisprudence, between legal formalism and legal scepticism. Legal formalism holds that legal decisions can and should be reached by a strictly logical chain of arguments using authoritative legal rules. Scepticism holds, in contrast, that legal decisions are made on the basis of personal value preferences and are merely rationalised by reference to precedents and legal rules.

The argumentation theory of this paper potentially reconciles these two conflicting views. It uses classical forms of logic to construct arguments which derive conclusions from given facts extended by means of assumptions. It evaluates alternative extensions and therefore the arguments they support, by comparing their relative ability to defend themselves against attack. It differs from popular conceptions of logic in that it does not aim to determine a unique collection of truths that follow deductively from a given set of facts and rules. It allows the possibility that several alternative, but mutually incompatible extensions and the arguments they support might be equally acceptable.

3 Theorist

Poole's Theorist [13] is probably the simplest example of the kind of argumentation system we consider in this paper. In Theorist there are two kinds of "beliefs":

- "facts", which are ordinary sentences of first-order logic, and
- "assumptions", which are formulae of first-order logic.

Default reasoning within a given consistent set of facts T (also called a "theory") is performed by constructing a maximally consistent **extension** E of T, where E is the set of all logical consequences of $T \cup \Delta$, and Δ consists of variable-free instances of the given candidate assumptions. An **argument** supporting a conclusion C is a deductive proof of C from E.

In general, there will be many different, mutually incompatible extensions of a given theory T. Different extensions can allow different arguments with conflicting conclusions that contradict one another.

Example 3.1 Consider the case of the wealthy businessman who makes a valid will leaving all his estate to his wicked grandson. The grandson, being wicked, murders the grandfather to inherit the estate. The grandson is arrested and sentenced to 15 years imprisonment. Does the grandson inherit the estate? Under a literal interpretation of the law, he does, because there is a valid will. Under a liberal interpretation, which takes into account the intention of the law, that no man profits from committing a crime, he does not.

The case, greatly simplified, can be formalised in the theorist framework by means of the theory

inherit if
 valid-will and literal-interpretation
not inherit if
 murder and liberal-interpretation
valid-will
murder

where the only candidate assumptions are the predicates

literal-interpretation, and
liberal-interpretation.

The case is a "hard" one, because there are two mutually incompatible, maximally consistent extensions. In one extension, containing the first assumption, the grandson inherits the estate. In the other, containing the second assumption, he does not. It is possible, of course, to argue on other grounds that the second extension and the conclusion it supports is preferable to the first.

The identification of candidate assumptions to use for extending a given theory is critical to the Theorist approach and to argumentation in general. One common criterion, explored implicitly or explicitly in many approaches to default reasoning is to treat all negations of atomic predicates as candidate assumptions. This is the underlying convention in relational databases and logic programming, in particular, where a negative statement, *not p*, is deemed to hold if its positive contrary p, can not be shown to hold.

Example 3.2 Consider, for example, the simplified theory consisting of the following "facts":

X is innocent if not X is guilty
X is guilty if X confesses and X has a motive
john has a motive.

Assume that the candidate assumptions are all the (variable-free) instances of the negative predicates

not X is innocent

$$not\, X\, is\, guilty$$
$$not\, X\, confesses$$
$$not\, X\, has\, a\, motive$$

and consider only those instances in which X is *john*. Perhaps somewhat surprisingly, there are two maximally consistent extensions. Both contain the assumption

$$not\, john\, confesses.$$

One contains the assumption

$$not\, john\, is\, guilty$$

which supports the conclusion that john is innocent. The other contains the assumption

$$not\, john\, is\, innocent$$

which supports the conclusion that john is guilty. Clearly the second conclusion and the extension that justifies it are not in accord with the intended understanding of the first sentence in the theory as meaning that

> a person is innocent if
> the person is not proven to be guilty.

To eliminate the undesired extension, different logics for default reasoning use different techniques.

Theorist uses integrity constraints to prevent the use of contrapositives. Logic programming and default logic also prevent the use of contrapositives: logic programming by restricting all uses of "if" to the application of modus ponens; and default logic by allowing a choice between interpreting "if" as classical material implication, which allows contrapositive reasoning, and interpreting "if" as signalling a theory-specific inference rule, to which only modus ponens applies.

Autoepistemic logic and non-monotonic modal logic retain the interpretation of "if" as classical material implication, but interpret default negation

$$not\, p$$

in effect, as

$$\neg Lp$$

where L is a modal operator which can be understood as "it is believed that" or "it is proved that" and $\neg$ is classical negation. The candidate assumptions include all negative sentences of the form $\neg Lp$. Contrapositive reasoning is allowed for classical negation, but not for default negation. Thus given

$$p \leftarrow \neg Lq$$

we can derive teh contrapositive

$$Lq \leftarrow \neg p$$

but not

$$Lq \leftarrow \neg Lp.$$

Therefore, in particular, if we represent the belief that a person is innocent if not proved guilty by the sentence

$$X\, is\, innocent \leftarrow \neg L\, X\, is\, guilty$$

then we can derive the contrapositive

$$L\, X\, is\, guilty \leftarrow \neg X\, is\, innocent$$

but from the assumption

$$\neg LX\, is\, innocent$$

we can derive neither

$$\neg X\, is\, innocent$$

nor

$$L\, X\, is\, guilty$$

nor

$$X\, is\, guilty.$$

We shall return to autoepistemic logic and non-monotonic modal logic briefly later in this paper.

4 Logic Programming

Logic programs can be understood as argumentation systems in which, similar to Theorist, beliefs are of two kinds:

- "facts", which are rules of the form

  $p\ \ if\ \ q_1\ \ and\ \dots\ q_n\ \ and\ not\ \ r_1\ \ and\ \dots\ not\ \ r_m$

  where p, $q_1, \dots, q_n$, $r_1, \dots, r_m$ are all atomic formulae, $n \geq 0$ and $m \geq 0$, and

- "assumptions", which are all the negations of atomic predicates.

A variety of semantics have been defined for logic programs understood in these terms. Interestingly, none of these semantics is equivalent to the maximal consistency semantics of Theorist.

Example 4.1 Consider example 3.2 again. This has the syntax of a logic program and the same assumptions, consisting of all negative atomic predicates. As in the Theorist case, there are two maximally consistent extensions. Both contain the assumption

$$not\, john\, confesses.$$

One contains the assumption

$$not\, john\, is\, guilty$$

which supports the conclusion that john is innocent. The other contains the assumption

$$not\, john\, is\, innocent.$$

But, because the law of contrapositives does not hold, this second extension does not support the conclusion that john is guilty. Nonetheless, the second extension is undesirable. It is not allowed in any of the standard semantics of logic programming, all of which impose a more restrictive requirement on extensions than simple consistency. Perhaps the simplest of these is the stable model semantics [7].

Given an inconsistent extension and several ways of restoring consistency, the stable model semantics in effect restores consistency by removing assumptions which are attacked by the rest of the extension in preference to removing assumptions which are not so attacked. In general, an extension E **attacks** another extension E' if and only if E attacks some assumption α in E', and an extension E attacks an assumption α if and only if E contains the contrary of α. In logic programming the contrary of an assumption $not\, p$ is p. In general,

> An extension is **stable** if and only if it does not attack itself, but does attack every assumption which is not in the extension.

Stable extensions, therefore, classify all candidate assumptions into two kinds: "those who are with us" and "those we are against". In the case of logic programs, where the set of candidate assumptions is the set of all negations of atomic sentences, this classification can be understood as determining the "truth value" of every atomic sentence as either "true" or "false". Therefore, every stable extension for a given logic program determines a unique interpretation in which the program itself is evaluated as "true". This interpretation is called a **stable model**.

In the case of example 3.2, stable model semantics allows only the extension containing the two assumptions

$$not\ john\ confesses$$
$$not\ john\ is\ guilty$$

which attacks the two assumptions

$$not\ john\ has\ a\ motive$$
$$not\ john\ is\ innocent$$

which are not in the extension. The second extension containing

$$not\ john\ confesses$$
$$not\ john\ is\ innocent$$

is not stable because it does not attack the assumption

$$not\ john\ is\ guilty$$

which is not in the extension.

But stable model semantics is too restrictive, as the following variant of example 3.1 shows.

Example 4.2 Let the given logic program be

inherit if valid-will and
not liberal-interpretation
disinherit if murder and
not literal-interpretation
liberal-interpretation if
inherit and disinherit
literal-interpretation if
inherit and disinherit
valid-will
murder.

Here the positive assumptions of example 3.1 have been renamed as negative assumptions, and the negative conclusion of the second rule has been renamed as a positive predicate. The facts that inherit and disinherit are contradictory and that a contradiction implies any conclusion in classical logic are partially simulated by the third and fourth rules. The program has no stable extension. This is because any extension that contains both of the assumptions

$$\begin{array}{l} \textit{not liberal-interpretation} \\ \textit{not literal-interpretation} \end{array}$$

attacks itself. But the only way to attack either of these two assumptions is to derive both *inherit* and *disinherit*, which in turn requires the use of the same two assumptions.

Of course, there are two maximally consistent extensions, one containing

$$\begin{array}{l} \textit{not liberal-interpretation} \\ \textit{not disinherit} \end{array}$$

the other containing

$$\begin{array}{l} \textit{not literal-interpretation} \\ \textit{not inherit} \end{array}$$

neither one of which is stable.

There is, however, an alternative semantics which sanctions both of these extensions. In general,

> an extension E is **acceptable** if and only if it does not attack itself and, for every extension E' that attacks E,
> E defends itself against E'.

The notion of defence can be understood more or less liberally. In the admissibility semantics [3],

> E **defends** itself against E' if and only if E attacks E'.

In the stable theory semantics [8],

> E **defends** itself against E' if and only if the extension consisting of all logical consequences of $E \cup E'$ attacks the extension consisting of all logical consequences of $E' - E$.

The logic program of example 4.2 has no stable models and no acceptable extensions in the sense of the admissibility semantics. However, the two maximally consistent extensions are acceptable in the sense of the stable theory semantics.

In this example there are three additional acceptable extensions in the sense of the stable theory semantics, namely the extensions consisting of all logical consequences of the program augmented by the the empty set of assumptions, by the set

$$\{\textit{not liberal-interpretation}\}$$

and by the set

$$\{\textit{not literal-interpretation}\}$$

respectively. Neither of these extensions provides a "total" interpretation of the program in the sense of stable extensions, where each sentence is either "true" or "false". For instance, the third additional
extension contains neither *liberal-interpretation* nor *not liberal-interpretation*. Therefore, an acceptable extension need not take a stand on every issue.

This feature of the acceptability semantics facilitates the computation of acceptable extensions supporting a given conclusion. Given a program, the computation first uses the underlying monotonic logic to find an extension E_0 containing the given conclusion, and then generates an extension E containing E_0 such that E is acceptable. E is contructed incrementally in such a way that it defends E_0 against all attacks and it is acceptable.

For the logic program of example 4.2, the conclusion *disinherit* holds in the extension E_0 containing the assumption *not literal-interpretation*. However, E_0 is attacked by the extension E' containing the assumptions

$$\begin{array}{l} \textit{not literal-interpretation} \text{ and} \\ \textit{not liberal-interpretation}. \end{array}$$

Because the extension consisting of all logical consequences of $E_0 \cup E'$ attacks E', E_0 is acceptable in the sense of the stable theory semantics. Therefore, the computation returns the extension E given by $E_0 \cup \emptyset$. This procedure is a generalisation of the Eshghi-Kowalski procedure that computes admissible extensions for logic programming [5].

5 The Abstract Argumentation Theory

The terminology we have used for Theorist and logic programming in the previous section, to describe different argumentation-theoretic notions, can be used more generally for other non-monotonic logics. The definitions of extension, attack, defence, stable extension, acceptable extension (both in the sense of admissibility semantics and in the sense of stable theory semantics) apply to any theory formulated in any monotonic logic. Similarly, the proof procedure for computing acceptable extensions can also be applied more abstractly. In general, it is necessary only to identify

- the **underlying language** in which theories are formulated;
- the **candidate set of assumptions** that can be used to extend any theory;
- the notion of what it means to be the **contrary of an assumption**.

In the case of **Theorist** these are

- any first-order language;
- any set of sentences in the language;
- the notion that $\neg\alpha$ is the contrary of an assumption α.

In the case of **logic programming** they are

- the language of rules, defined in section 4;
- the set of negations of atomic sentences;
- the notion that p is the contrary of an assumption *not* p.

Default logic, autoepistemic logic and non-monotomic modal logic can be characterised similarly in argumentation-theoretic terms. The standard semantics of these logics can then be shown to be special cases of stable extension semantics in general [2].

In the case of **default logic**

- The language is any first-order language augmented with sentences of the form

  $$p \leftarrow q_1 \wedge \ldots \wedge q_n \wedge Mr_1 \wedge \ldots \wedge Mr_m$$

  where $p, q_1, \ldots q_n, r_1 \ldots r_m$ are all first-order formulae, $n \geq 0$, $m \geq 0$, $\leftarrow$ is a new logical symbol, for which only the rule of modus ponens applies, $\wedge$ stands for "and" and M is a new logical symbol not used elsewhere in the language.
- The set of candidate assumptions is the set of all sentences of the form Mr, where r is any sentence in the underlying language.
- The contrary of an assumption Mr is the sentence $\neg r$.

In the case of **autoepistemic logic**

- The language is any propositional first-order language with a modal operator L, where, however, the underlying semantics of the language is classical logic.
- The set of candidate assumptions is the set of all sentences of the form Lr or of the form $\neg Lr$, where r is any sentence of the underlying language.
- The contrary of an assumption Lr is $\neg Lr$. The contrary of an assumption $\neg Lr$ is r.

In the case of **non-monotonic modal logic**

- The language is any first-order language, with a modal operator L, where, differently from autoepistemic logic, the semantics of the language is modal logic, with the necessitation rule of inference:

  $$\frac{r}{Lr}.$$

- The set of candidate assumptions is the set of all sentences of the form $\neg Lr$, where r is any sentence of the underlying language.
- The contrary of an assumption $\neg Lr$ is r.

The argumentation-theoretic characterisation of these different logics clarifies their underlying similarities and differences. Some of these differences are relatively trivial. For example the assumptions *not* p in logic programming, Mr in default logic and $\neg Lr$ in auto-epistemic logic and non-monotonic modal logic can all be mapped into one another by syntactic renaming of positive expressions as negative expressions and vice versa.

The argumentation theory also shows that, despite their differences, all of these logics can be understood in the same terms, as sanctioning an extension if and only if it is stable. As we have seen in example 4.2, stable semantics is too restrictive for logic programming. However, the same example can also be formulated in each of the other logics, and shows, therefore, that stable semantics is too restrictive in general. It also shows that the notion of acceptable extension (especially in the sense of stable theory semantics) is generally preferable to stable semantics.

A number of other argumentation-theoretic formalisms for non-monotonic reasoning have been proposed. These include the formalisms of Dung [4], Pollock [12], Simari and Loui [15] and Geffner [6]. Dung's formalism [4] differs from ours in the higher level of abstraction with which it treats the notions of assumptions, arguments and attacks. The other three formalisms differ from ours both in their being more concrete and in their justifying sceptical rather than credulous forms of non-monotonic reasoning.

All of the semantics we have considered until now are credulous in the sense that they justify a conclusion as a non-monotonic consequence of a given theory if and only if it holds in at least one extension sanctioned by the semantics. Sceptical semantics, on the other hand, justifies a conclusion if and only if, in some sense, it belongs to the common ground on which all credulous semantics agree. The argumentation theory can also be used to define a sceptical semantics in general.

6 Sceptical Semantics

The sceptical semantics justifies holding a conclusion if and only if it belongs to the smallest extension which does not attack itself and contains every assumption it can defend. This extension is called the **grounded** extension. As in the case of acceptable extensions, the notion of defence can be understood in different ways. If we use the notion of defence in the sense of the admissibility semantics, then the grounded extension corresponds to the well-founded model in logic programming [17].

In example 3.2, the unique grounded extension contains the two assumptions

$$\begin{array}{c} not\ john\ confesses \\ not\ john\ is\ guilty \end{array}$$

neither of which is attacked by any set of assumptions.

In example 4.2, the unique grounded extension is the set of all logical consequences of the program augmented by the empty set of assumptions.

7 A Relationship with Belief Revision

The argumentation-theoretic approach also allows us to define other semantics. We have already seen that we can define different notions of defence for the acceptability semantics. Similarly, we can define different notions of attack. For example, in the spirit of stable theory semantics, we can say that

> an extension E **attacks** another extension E' if and only if the extension consisting of all logical consequences of $E \cup E'$ contains the contrary of some assumption α in E'.

This new notion of attack has a natural reinterpretation in belief revision terms:

> an extension E **attacks** another extension E' if and only if the extension consisting of all logical consequences of $E \cup E'$ contains a **conflict** (in the form of an assumption α and its contrary) which can be removed by removing α in E'.

The notions of conflict and of removing a conflict, in the belief revision notion of attack, can be defined, without the notion of contrariness, in terms of **integrity constraints with retractibles** [16]. Namely, a conflict between an assumption α and its contrary $\overline{\alpha}$, which can be removed by removing α, can be represented as a violation of the integrity constraint

$$\neg[\alpha \wedge \overline{\alpha}]$$

in which α has been identified as retractible.

Integrity constraints can also be used to define more general notions of attack, as in the case of abductive logic programming [16]. Another use of integrity constraints will be illustrated in the next section.

8 Conflict Resolution

We have used the argumentation theory, until now, to formalise different ways in which an agent can use assumptions to justify its beliefs. In particular, we have seen that an agent can "aggressively" take a stand on every issue (stable semantics), "liberally" hold a belief by defending it against all possible attacks (acceptability semantics), or "cautiously" hold only those beliefs that are also held by every other agent (grounded semantics). In each of these cases the agent justifies its beliefs by attacking other beliefs held by other, hypothetical agents. Another important, but more difficult case is the one in which the other agents are real and the goal is to reconcile conflicts between the different agents. In this section we outline an initial proposal for an argumentation-theoretic approach to such conflict resolution.

Suppose that two agents hold conflicting beliefs which represent different conflicting actions they intend to carry out in the future. Our proposal is first to try to identify a set of goals and shared beliefs upon which the two agents can agree and then to try to find a solution of the shared goals which is compatible with the shared beliefs. The first of these two steps is the most important and requires the greater creativity. Typically, it involves identifying the agents' possibly conflicting goals and generalising them to a more abstract level where they no longer conflict.

In the simplest case, the agents' goals might not conflcit at all and there is a solution to the combined goals, alternative to the original conflicting solutions, which is acceptable to both agents. In other cases, the original goals may need to be generalised before a shared solution can be found. The following case study illustrates this second, more typical case.

In a recent head-of-sections committee meeting in our Department, we discussed the composition of a new resources committee. Two conflicting arguments were put forward. The Director of Administration argued that, in the interests of efficiency, the members of the new committee should consist of himself and the other principal administrative officers of the Department. The Director of Research argued, in opposition to him, that, in the interests of democracy, the committee should also contain members elected by the Department. During the course of the discussion it became clear that the two sides were focussing on different assumptions about the purpose of the new committee: the Director of Administration on its purely administrative function, and the Director of Research on its presumed policy making nature. These two assumptions could be viewed as conflicting solutions to the more general goals of deciding, on the one hand, which group should administer resources, and on the other hand, which group should make policy about resources.

By focussing on the more general goals, it was possible to identify a new solution which was acceptable to both parties: the resources committee will administer resources, whereas the head-of-sections committee will make policy about resources. In the interests of efficiency, the members of the resources committee will consist of administrative officers only. In the interests of democracy, the head-of-sections committee will represent the views and interests of the various Department sections on matters concerning policy about the allocation of resources.

The process of reconciliation can be rationally reconstructed more formally:

Original goal

$$composition\ of\ res\text{-}c\ \ is\ of\ type\ X.$$

Original candidate assumptions

$$composition\ of\ res\text{-}c\ \ is\ of\ type\ non\text{-}elected$$
$$composition\ of\ res\text{-}c\ \ is\ of\ type\ elected.$$

Each assumption attacks the other.

Solution one

$$composition\ of\ res\text{-}c\ \ is\ of\ type\ non\text{-}elected.$$

This is "supported" by the additional assumption

$$res\text{-}c\ \ administers\ resources$$

by the integrity constraint

$$X\ is\ efficient\ \ if\ \ X\ administers\ Y$$

and by the rule

$$\begin{aligned}&X\ is\ efficient\ \ if\\ &\quad composition\ of\ X\ is\ of\ type\ non\text{-}elected.\end{aligned}$$

Notice that the integrity constraint expresses a property that *should* be satisfied *independently* of the integrity constraint: if an entity administers something then that entity *should* be efficient. The *obligation* of efficiency, however, needs to be satisfied by some means other than the integrity constraint. In this case, the rule expresses one such way. Presumably, another way might be to have no committee at all.

Solution two

$$composition\ of\ res\text{-}c\ \ is\ of\ type\ elected.$$

This is supported by the additional assumption

res-c makes policy about resources

by the integrity constraint

X is democratic if X makes policy about Y

and by the rule

X is democratic if
composition of X is of type elected.

Another way of achieving democracy is expressed by the additional rule

X is democratic if
composition of X is of type representative.

Refined goals

X administers resources and
composition of X is of type Y and
U makes policy about resources and
composition of U is of type V.

These goals generalise the original explicitly formulated goal, as well as the original implicit goals of the two agents.

Refined solution

res-c administers resources
composition of res-c is of type non-elected
h-of-s-c makes policy about resources.

These three assumptions solve the first three subgoals. The fourth goal is solved by the fact

composition of h-of-s-c is of type representative.

The refined solution achieves the refined goals of both, originally conflicting agents. It builds upon the fact that each agent accepts the other agent's integrity constraints and rules. It relies upon each agent's willingness to entertain the other agent's goals and to agree upon a refined set of goals, which takes the two original, different sets of goals into account. It also relies upon the second agent's willingness to agree upon a different solution from the one he originally proposed.

9 Conclusion

In this paper we have sketched a theory of argumentation which has proved useful for unifying and generalising different approaches to default reasoning. We have observed that the standard, stability semantics of most approaches to default reasoning is undesirably and unnecessarily intolerant; whereas admissibility and stable theory semantics, which were first proposed as semantics for logic programming, are more appropriate semantics for practical reasoning in general. We have also proposed an extension of the argumentation theory to a theory of conflict resolution. We hope that this initial proposal might eventually provide the basis for a practical and systematic approach to the reconciliation of conflicts in the future.

Acknowledgements

This research was supported by Fujitsu Research Laboratories. The authors are grateful to Phan Minh Dung for many helpful discussions.

References

[1] A. Bondarenko, F. Toni, R. A. Kowalski, An assumption-based framework for non-monotonic reasoning. *Proc. 2nd International Workshop on Logic Programming and Non-monotonic Reasoning,* (A. Nerod and L. Pereira eds.) MIT Press (1993)

[2] A. Bondarenko, P. M. Dung, R. A. Kowalski, F. Toni, An abstract, argumentation-theoretic framework for default reasoning. In preparation (1994)

[3] P. M. Dung, Negation as hypothesis: an abductive foundation for logic programming. *Proc. 8th International Conference on Logic Programming,* Paris, MIT Press (1991)

[4] P.M. Dung, The acceptability of arguments and its fundamental role in non-monotonic reasoning and logic programming. *Proc. IJCAI'93* (1993)

[5] K. Eshghi, R.A. Kowalski, Abduction compared with negation as failure. *Proc. 6th International Conference on Logic Programming,* Lisbon, Portugal, MIT Press (1989)

[6] H. Geffner, Beyond negation as failure. *Proc. 2nd International Conference on Principles of Knowledge Representation and Reasoning,* Cambridge, Mass. (1991) 218–229

[7] M. Gelfond, V. Lifschitz, The stable model semantics for logic programming. *Proc. 5th International Conference and Symposium on Logic Programming,* Washington, Seattle, MIT Press (1988)

[8] A. C. Kakas, P. Mancarella, Stable theories for logic programs. *Proc. ISLP'91,* San Diego (1991)

[9] A. C. Kakas, R. A. Kowalski, F. Toni, Abductive logic programming. *Journal of Logic and Computation* 2(6) (1993)

[10] D. McDermott, Nonmonotonic logic II: non-monotonic modal theories. *JACM* 29(1) (1982)

[11] R. Moore, Semantical considerations on non-monotonic logic. *Artificial Intelligence* 25 (1985)

[12] J.L. Pollock, Defeasible reasoning. *Cognitive Science,* Vol. 11 (1987) 481–518

[13] D. Poole, A logical framework for default reasoning. *Artificial Intelligence* 36 (1988)

[14] R. Reiter, A logic for default reasoning. *Artificial Intelligence* 13 (1980)

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

[16] F. Toni, R.A. Kowalski, Reduction of abductive logic programs to normal logic programs. Imperial College Technical Report, London (1994)

[17] A. Van Gelder, K.A. Ross, J.S. Schlipf, Unfounded sets and the well-founded semantics for general logic programs. *Proc. ACM SIGMOD-SIGACT, Symposium on Principles of Database Systems* (1988)