# Extending the Abstract Argumentation Framework to Include Argument Dynamics


**Daniela Carbogim*** **Dave Robertson*** **John Lee†**

* Institute for Representation and Reasoning - Division of Informatics - University of Edinburgh
80 South Bridge - Edinburgh, Scotland - EH1 1HN {danielac,dr}@dai.ed.ac.uk
† Human Communication Research Centre - Division of Informatics - University of Edinburgh
2 Buccleuch Place - Edinburgh, Scotland - EH8 9LW john@cogsci.ed.ac.uk



### Abstract

The Abstract Argumentation Framework gives a flexible way of dealing with defeasibility in argument, but it lacks features necessary to describe the dynamics of argument. Here we present an extension of the original framework and demonstrate its use on an example from a standard safety engineering reference text.



**Keywords:** argumentation, logic programming, safety-critical domains.


## The Abstract Argumentation Framework

In this section we outline the Abstract Argumentation Framework presented in (Kowalski & Toni 1996; Bodarenko *et al.* 1997).

This 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. This system defines the following provability relation: $Th \vdash \alpha$ if there is a *deduction* of $\alpha \in \mathcal{L}$ from a theory $Th$. A theory is any set $Th \subseteq \mathcal{L}$.

**Definition 1** *Let $(\mathcal{L}, R)$ be a monotonic deductive system. An Abstract Argumentation Framework w.r.t. $(\mathcal{L}, R)$ is defined by:*

- *a theory $Th \subseteq \mathcal{L}$ representing facts or beliefs;*
- *a set of assumptions $\mathcal{A} \subseteq \mathcal{L}$, $\mathcal{A} \neq \emptyset$, that can extend any theory; and*
- *the notion of the contrary of an assumption $\alpha \in \mathcal{A}$, represented here by $\overline{\alpha} \in \mathcal{L}$.*

A key motivation for this framework is that it is 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 of their contrary. 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 & Toni 1996).

Defeasible reasoning can be formalised by extending $\mathcal{L}$ with a non-provability operator that expresses sentences of the form $cannot\_be\_shown(\alpha)$, where $\alpha \in \mathcal{L}$. Let $\mathcal{A}$ be the set of these sentences. Next we extend the rules in $R$ appropriately by adding assumptions as extra premises, thus reducing the defeasibility of the rules to the assumptions themselves. Note that $\overline{cannot\_be\_shown(\alpha)} = \alpha$.

**Definition 2** *If a conclusion $\alpha \in \mathcal{L}$ can be derived from $\Delta \subseteq \mathcal{A}$ and $Th \subseteq \mathcal{L}$, then we say that $Th \cup \Delta \vdash \alpha$ is an argument for $\alpha$.*

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

**Definition 3** *Given a theory $Th$, $\Delta \subseteq \mathcal{A}$ attacks $\Delta' \subseteq \mathcal{A}$ if there is $\alpha \in \Delta'$ such that $Th \cup \Delta \vdash \overline{\alpha}$.*

Arguments can be evaluated in terms of their ability to defend themselves against attack (Kowalski & Toni 1994). For instance, we can define the class of *acceptable arguments* as follows:

**Definition 4** *An argument $Th \cup \Delta \vdash \alpha$ is acceptable if and only if the set of assumptions $\Delta$ on which it is based is admissible.*

**Definition 5** *A set of assumptions $\Delta \subseteq \mathcal{A}$ is admissible if and only if, for every $\Delta' \subseteq \mathcal{A} - \Delta$, if $\Delta'$ attacks $\Delta$ then $\Delta$ attacks $\Delta'$.*

Because the defeasibility of the rules is reduced to the non-provability premises, all relations between arguments are reduced to undermining attacks. Priorities between arguments can also be defined in terms of non-provability assumptions.

### Some Limitations

In (Kowalski & Toni 1996), it is claimed that the Abstract Argumentation Framework "seems to correspond well with informal argumentation". However, there are some informal arguments that cannot be represented in this framework.

Consider, for example, a system that models the operation of the pressure tank control system in figure 1, as defined in the Fault Tree Handbook (U.S 1981):

> "The pump pumps fluid from an infinitely large reservoir into the tank. We shall assume that it takes 60 seconds to pressurize the tank. The pressure switch has contacts which are closed when the tank is empty. When the threshold pressure has been reached, the pressure switch contacts open, deenergizing the coil of relay K2 so that relay K2 contacts open, removing power from the pump, causing the motor to cease operation. The tank is fitted with an outlet valve that drains the entire tank in an essentially negligible time. [...] When the tank is empty, the pressure switch contacts close, and the cycle is repeated."

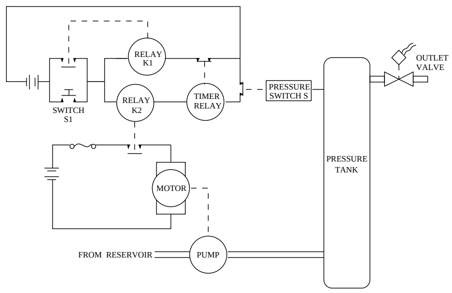


Figure 1: The pressure tank system

Formal arguments for the safety of this system may involve a proof that the system is operational at all times. In safety-critical domains, however, these arguments might still be evaluated in terms of fault tree analysis.

A fault tree is a model of what faults can lead to an unsafe event, or top event. Fault tree analysis evaluates weaknesses of the system by assessing the fault tree qualitatively and quantitatively. It identifies the possible combinations of basic events in a fault tree from which the top event can be derived (namely *minimal cut sets*), and estimates the probability of the top event from the probabilities assigned to the basic events. In this way, fault tree analysis is not only a source of possible arguments against the safety of a system, but it also provides criteria for priority and relevancy of such arguments.

Consider the top event of a fault tree for this system to be the *"rupture of pressure tank after the start of pumping"*[1]. One of the minimal cut sets of this fault tree is composed of the basic event *"primary failure of K2"*. By primary failure of a component we mean that the component fails to work under circumstances on which it should work. In this case, if K2 contacts fails to open when the coil of K2 is deenergised, then the tank will rupture.

The importance of this argument can be measured through fault tree analysis by taking the ratio of its probability to the estimated probability of the top event. For example, the minimal cut set composed of *"primary failure of K2"* has relative quantitative importance of 86% w.r.t. the derivation of the top event. This is a strong argument, but it can be undermined if we add some redundancy to the system; i.e. system safety could be considerably improved by adding another relay in parallel to K2.

Here we are interested in formalising the dynamics of argumentation in the sense described above, where an argument is attacked by another one but it can defend itself and reestablish its validity by undermining this attack. In particular, consider the argument about the safety of the pressure tank system.

To represent this argument in terms of an Abstract Argumentation Framework we first select a supporting deductive system. For instance, the Horn clause resolution-based system of (Kowalski & Toni 1996). Let $\mathcal{A}$ be the set of assumptions of the form $cannot\_be\_shown(\alpha)$, where $\alpha$ is a sentence in the underlying language. Assuming that we know how to extend the rules appropriately, the following is a (fragment of a) model of the pressure tank system in figure 1.

$$\begin{array}{l} operational\_tank(T) \leftarrow \\ \quad on\_motor(T) \wedge not\_full(T) \end{array}$$

$$\begin{array}{l} operational\_tank(T) \leftarrow \\ \quad off\_motor(T) \wedge pressurised(T) \end{array}$$

$$\begin{array}{l} on\_motor(T) \leftarrow \\ \quad closed(rcontacts(k2), T) \end{array}$$

$$\begin{array}{l} off\_motor(T) \leftarrow \\ \quad open(rcontacts(k2), T) \end{array}$$

$$\begin{array}{l} closed(rcontacts(K), T) \leftarrow \\ \quad energised(rcoil(K), T) \wedge \\ \quad cannot\_be\_shown(open(rcontacts(K), T)) \end{array}$$

$$\begin{array}{l} open(rcontacts(K), T) \leftarrow \\ \quad not\_energised(rcoil(K), T) \wedge \\ \quad cannot\_be\_shown(closed(rcontacts(K), T)) \end{array}$$

Briefly, the tank is operational at a time point $T$ if the motor is pumping water into it when it is not full, or when the tank is pressurised but the motor is off. However, it is possible that relay K2 contacts fail to open when the coil is deenergised, causing the rupture of the tank. We represent this fault by adding the following clauses to the model, where $tank\_rupture(T)$ contradicts $operational\_tank(T)$:

$$\begin{array}{l} tank\_rupture(T) \leftarrow \\ \quad on\_motor(T) \wedge pressurised(T) \end{array} \tag{1}$$

$$\begin{array}{l} closed(rcontacts(k2), T) \leftarrow \\ \quad not\_energised(rcoil(k2), T) \end{array} \tag{2}$$

[1]See chapter VIII of the Fault Tree Handbook (U.S 1981) for the fault tree analysis w.r.t. this top event.

Because the underlying system is monotonic, the addition of these clauses only allows more possible conclusions to be derived. In fact, let

$Th = \{not\_energised(rcoil(k2), T), pressurised(T)\}$,
$\Delta = \{cannot\_be\_shown(closed(rcontacts(k2), T))\}$.

We then have two conflicting arguments:

$$Th \vdash tank\_rupture(T) \text{ and}$$
$$Th \cup \Delta \vdash operational\_tank(T).$$

Note that the second argument cannot defend itself against $Th \vdash closed(rcontacts(k2), T)$. Hence, on the basis of acceptable arguments (or admissible assumptions) as defined by Kowalski & Toni, this framework discriminates among faulty and non-faulty behaviours and allows only the inference of $tank\_rupture(T)$.

This is an important point because it shows that the Abstract Argumentation Framework can formalise part of the safety argument about the pressure tank system. In this particular case where a fault is present, it does not allow a safe conclusion to be erroneously derived.

Yet, this is a matter of system safety, so it is essential that we can adapt the model so that it only exhibits safe behaviours. In this case, we could add more assumptions to $\Delta$ or further extend the rules in $R$, thus:

- defending the arguments for $operational\_tank$ against all possible attacks; and
- prioritising these arguments over the arguments for $tank\_rupture$.

These solutions, however, do not represent any enhancement of system safety. Arguments such as *"if an extra relay is added in parallel this attack will no longer be relevant enough to be a concern"* – which are common when modelling systems in safety-critical domains – cannot be expressed by means of extra non-provability assumptions.

Although the Abstract Argumentation Framework is effective in expressing the defeasibility in argument, it does not account for many of the features which are responsible for the dynamics of argument. In particular, it does not describe where attacks on an argument (e.g. expressions (1) and (2) above) come from; how the source of these attacks maps onto changes to the argument; how these lead to counter-attacks; and how we determine the comparative strength of criticism.

## A General Architecture for a Formal Argumentation Framework

In this section we introduce a general architecture for a formal argumentation framework, which identifies the different notions involved in an argument and defines them separately. This approach provides a clearer picture of how arguments can be generated and evaluated. It also allows a larger class of arguments to be formally represented, in particular the kind of argument described in the previous section.

To define this architecture, we need to identify three parts, namely:

- a theory;
- a criticism theory; and
- a control module.

### The Theory

The notion of a theory here is broader than in the Abstract Argumentation Framework. By theory we mean a proof system $(\mathcal{L}, R)$ that allows conclusions to be derived via the inference rules in $R$. In particular, let us assume that $(\mathcal{L}, R)$ is a Horn-clause resolution-based system, and let us denote by $\vdash$ the provability relation defined by it.

In this architecture, a theory $Th$ simply defines what we argue about. For instance, it might be a model of a system (e.g. the pressure tank system in figure 1) or of some knowledge we want to represent. Hence, this is the level where arguments are generated.

**Definition 6** *If a conclusion $\alpha \in \mathcal{L}$ can be derived from a theory $Th$, then $Th \vdash \alpha$ is an argument for $\alpha$.*

At this level we also define the notion of conflicting conclusions in terms of a special predicate $conflicts$. Consequently, we can define what it means for an argument to attack another one.

**Definition 7** *We say that an argument $Th \vdash \alpha$ attacks an argument $Th \vdash \beta$ if $conflicts(\alpha, \beta)$.*

However, to formalise argumentation it is not enough to characterise *what* is an argument. We need also to define *when* arguments are generated and *how* they are connected, and this requires an additional theory.

### The Criticism Theory

In the Abstract Argumentation Framework it is not always clear how to extend rules appropriately. By appropriately we mean that the non-provability assumptions which are added as extra premises explicitly represent the defeasibility of these rules. Such premises are assumed to be valid unless contrary evidence can be derived and, in this sense, they suggest which are the possible attacks to the arguments in the framework.

Our idea is to provide such potential points of attack within the architecture, rather than leaving it to the ingenuity of the arguer. Here, these possible attacks are defined separately from the theory itself, by means of an additional theory $T_{crit}$.

A criticism theory $T_{crit}$ is a Horn clause based proof system $(\mathcal{L}_{crit}, R_{crit})$. The provability relation $\vdash_{crit}$ defined by it does not have to be deductive. In fact, it can be abductive, as in the following situation.

Consider again the example presented in the previous section, where arguments relate to safety properties of the pressure tank system model. A candidate theory for $T_{crit}$ in that case is the fault tree model associated to that system. Fault trees are basically composed of *and/or* gates, and therefore can be easily expressed in a declarative way in terms of Horn clauses.

The following is a fragment of the fault tree model developed in (U.S 1981) and presented in this declarative style. The top event is represented by $tank\_rupture(T)$.

$$
\begin{aligned}
& tank\_rupture(T) \leftarrow \\
& \quad continuous\_pump\_operation(Ti, T) \wedge \\
& \quad duration(Ti, T, D) \wedge \\
& \quad greater(T, 60)
\end{aligned}
$$

$$
\begin{aligned}
& continuous\_pump\_operation(Ti, T) \leftarrow \\
& \quad fails\_to\_open(relay(k2), Ti, T)
\end{aligned}
$$

In short, the tank will rupture if the pump continuously operates for more than 60 seconds. This might happen if the relay K2 contacts fail to open during this time interval.

Let $\mathcal{A}_{crit} \subseteq \mathcal{L}_{crit}$ be the set of possible assumptions, or abducibles, in $T_{crit}$. In this case, $\mathcal{A}_{crit}$ corresponds to the set of basic events in the fault tree. In order to derive the top event from this criticism theory, we abductively select a subset of assumptions $\Delta \subseteq \mathcal{A}_{crit}$ such that $T_{crit} \cup \Delta \vdash_{crit} tank\_rupture(T)$. For instance,

$$\Delta = \{fails\_to\_open(relay(k2), Ti, T)\}.$$

Note that the notion of identifying minimal cut sets in fault trees is equivalent to applying abduction with minimality constraints to the corresponding declarative model. I.e., $\vdash_{crit}$ can be defined as abduction generating minimal sets of assumptions. This differs from the deductive inference used to determine consequences within the theory, $Th$.

Thus, $T_{crit}$ provides points of attack on arguments in $Th$ based on a set of abductively selected assumptions $\Delta \subseteq \mathcal{L}_{crit}$. Intuitively, each assumption is mapped into sets of clauses in $Th$ that allow an attack (or conflicting conclusion) to be derived. This mapping is twofold, as represented in figure 2:

- First we map conclusions in $Th$ to conclusions in $T_{crit}$ via a function $equiv : L \subseteq \mathcal{L} \rightarrow \mathcal{L}_{crit}$. This identifies which conclusions in $Th$ are subject to criticisms from $T_{crit}$.
- Next we map assumptions in $T_{crit}$ to sets of clauses in $Th$ through a function $map : \mathcal{A}_{crit} \rightarrow 2^{\mathcal{L}}$. This determines which clauses are to be added in the theory in order to derive the corresponding attack provided by the criticism theory.

Next we describe how the theory of criticism can generate an attack on a given argument

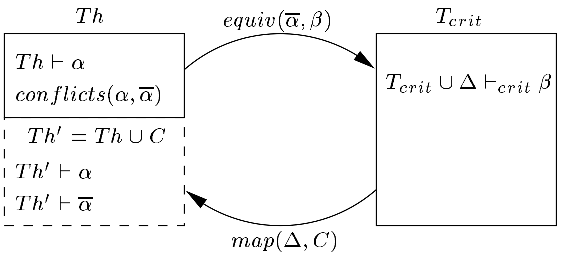


Figure 2: Generating attacks based on a criticism theory.

$Th \vdash \alpha$ in the theory. This process is also illustrated in figure 2:

- Let $\overline{\alpha} \in \mathcal{L}$ such that $conflicts(\alpha, \overline{\alpha})$.
- If $equiv(\overline{\alpha}) = \beta$, where $\beta \in \mathcal{L}_{crit}$, then $\beta$ is a possible point of attack on $\alpha$.
- Let $\Delta \subseteq \mathcal{A}_{crit}$ such that $T_{crit} \cup \Delta \vdash_{crit} \beta$. Then, $\Delta$ is the subset of assumptions on which the attack will be based.
- If $\Delta = \{\delta_i | 1 \leq i \leq n\}$ and $C_i = map(\delta_i)$ for $1 \leq i \leq n$, then $C = \bigcup_i^n C_i$ is the set of clauses that need to be added to the theory so that we can derive an argument for $\overline{\alpha}$ from it.
- Consider the extended theory $Th' = Th \cup C$. Because $Th$ is a monotonic deductive system, the addition of new clauses yields new consequences without compromising the original ones. Therefore, we now have that:

    i. $Th' \vdash \alpha$ and
    ii. $Th' \vdash \overline{\alpha}$.

  According to definition 7, (ii) is an argument that attacks (i).

In figure 3, we show how this process is instantiated for a particular argument about the safety of the pressure tank system. $Th$ and $T_{crit}$ are fragments of pressure tank system model and the associated fault tree model, respectively. In this case, from $T_{crit}$ we can derive an argument $Th' \vdash not\_operational\_tank(60)$ that attacks the safety argument $Th' \vdash operational\_tank(60)$.

In this way, a criticism theory can be considered as a source of attacks on arguments in the theory. Note that it is not a mere partition of the theory itself, as it may involve different inference mechanisms such as abduction. The last step in describing a general framework for formal argumentation is to define the notion of priorities between arguments.

## The Control Module

In human argument, it is often the case that extra information is applied to guide the process and assess the relevance of presented arguments. The

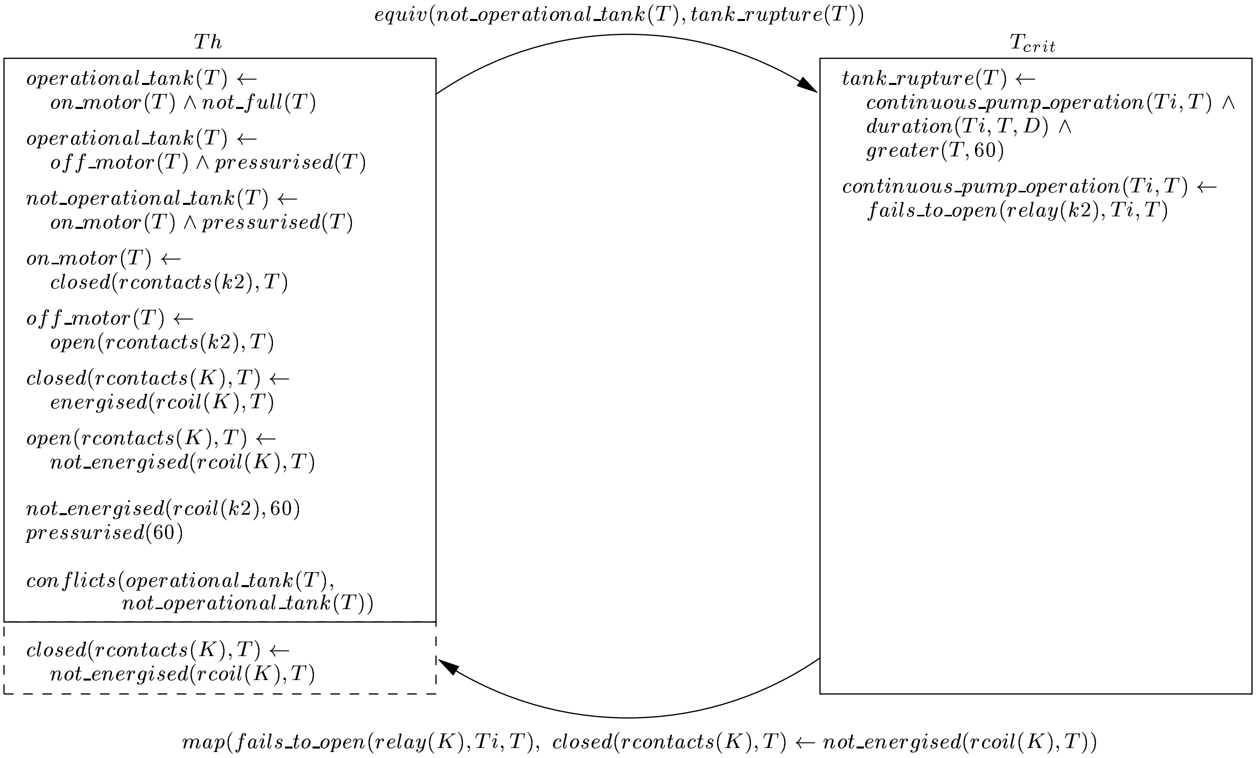


Figure 3: Generating attacks on the pressure tanks system model based on the fault tree model.

role of the control module is to define the extra information necessary to control the generation and termination of arguments within the framework.

As an illustration, let us look again at the safety argument regarding the pressure tank system in figure 1. Attacks on *operational_tank* can be generated by means of the criticism theory. Such attacks are based on subsets of assumptions that allow the top event in that theory to be derived. In the particular case of the pressure tank system, where the criticism theory is instantiated to the associated fault tree model, such subsets correspond to the notion of minimal cut sets.

As mentioned in the previous section, fault tree analysis provides means to assess the quantitative importance of minimal cut sets based on probabilities assigned to the basic events (or abducible sentences) in the fault tree model. For instance, from the analysis presented in (U.S 1981), we have that the quantitative importance of the minimal cut set composed of *"primary failure of K2"* is quite high (about 86%), meaning that its influence to the derivation of an unsafe event cannot be ignored. Therefore, some action should be done in order to lower such relative importance; for instance, adding an extra relay in parallel to K2. On the other hand, some minimal cut sets may not contribute in such high proportion to the occurrence of the top event. In these cases, we could safely ignore the corresponding attack and stick to the argument that derives a safe condition.

In this sense, we are applying some measures (probabilistic, in this particular case) to minimal cut sets. Such measures define the concept of relevance, or priority, of the corresponding attacks in relation to the attacked argument. Therefore, they also provide necessary information for deciding upon the termination of dynamic arguments.

The control module can then be defined as a PROLOG-style meta-interpreter for the criticism theory $T_{crit}$, which assigns measures to the assumptions $A_{crit} \subseteq \mathcal{L}_{crit}$ and propagate such measures through the rules in $T_{crit}$ appropriately. This architecture is similar to the form of layered meta-interpreter described in (Yalcinalp & Sterling 1991).

In general, the language of the control module comprises sentences of the form $measure(\delta, N)$, where $\delta \in A_{crit}$ and $N$ is a value expressing a choice of measure representation. These values are propagated according to this representation. For instance, in the case of the fault tree analysis, such values correspond to probabilities, which are distributed according to probability theory.

The criteria for priority of arguments is domain specific and depends of the choice of representation for these values. In this example, this criteria is the quantitative importance of the minimal cut sets in relation to the derivation of the top event. We could define a threshold importance value for minimal cut sets under which attacks based on such sets could be disregarded, and therefore the argument could be terminated.

Having defined these three levels of our framework, we can now outline how dynamic arguments could be generated from it.

### Generating an Argument

Suppose that we have defined an architecture for a formal argumentation framework according to the presentation above. Intuitively, a dynamic argument concerning the safety of the pressure tank system could be generated by this framework in the way described below:

1. Generate an argument for *operational_tank* in the theory.
2. Verify if an attack on this argument can be provided by the criticism theory (in this case, the fault tree model). As we have seen in figure 3, an argument for *tank_rupture* that attacks *operational_tank* can be generated based on the *"primary failure of relay K2"*.
3. Based on the priority criteria defined in the control module (probability theory and relative quantitative importance of minimal cut sets), this is a strong argument that compromises the safety of the system. Hence, it should be undermined.
4. To undermine this attack, add a relay in parallel to K2 and change the model of the system accordingly. System safety is reestablished; i.e. the argument for *operational_tank* is revalidated.
5. Revise the criticism theory and the control module to include the alteration. Repeat from step 2. This method has to be recursive because we might provide new points of attacks when revising the criticism.

Notice that we have not described how to change the theory to include structural modifications, nor how to revise the criticism theory and the control module. There are ways to define such activities systematically, but this is outside the scope of the architecture definition and will not be discussed in this paper.

## Conclusion

We take as a starting point the Abstract Argumentation Framework, which we applied to an example of argument from the safety engineering domain. Our first contribution is to show that at the level of the argument itself (corresponding in our example to the pressure tank system model) the existing framework accounts for plausible argument revision. Our second contribution is to identify questions which the existing framework does not answer. In particular:

- where attacks come from;
- when arguments terminate;
- how do we prioritise arguments.

We have presented an extended framework which takes these issues into account and have shown how it can be instantiated to our safety example.

**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 second author is supported by an EPSRC Advanced Fellowship. The third author is supported by HCRC, an Interdisciplinary Research Centre of the UK Economic and Social Research Council.

## References

Bodarenko, A.; Dung, P. M.; Kowalski, R. A.; and Toni, F. 1997. An Abstract, Argumentation-Theoretic Approach to Default Reasoning. *Artificial Intelligence* 93(1-2).

Kowalski, R., and Toni, F. 1994. Argument and Reconciliation. In *International Symposium on Fifth Generation Computer Systems '94, Workshop on Legal Reasoning*.

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

U.S. Nuclear Regulatory Commission. 1981. *Fault Tree Handbook (NUREG-0492)*.

Yalcinalp, L. U., and Sterling, L. 1991. Uncertainty Reasoning in Prolog with Layered Meta-Interpreters. In *Proceedings of the 7th IEEE Conference in AI Applications*, 398–402.