Knowledgebase Compilation for Efficient Logical Argumentation

Philippe Besnard
IRIT-CNRS,
Université Paul Sabatier,
118 rte de Narbonne, 31062 Toulouse, France

Anthony Hunter
Department of Computer Science,
University College London,
Gower Street, London, WC1E 6BT, UK

Abstract

There are a number of frameworks for modelling argumentation in logic. They incorporate a formal representation of individual arguments and techniques for comparing conflicting arguments. A common assumption for logic-based argumentation is that an argument is a pair $\langle\Phi, \alpha\rangle$ where Φ is minimal subset of the knowledgebase such that Φ is consistent and Φ entails the claim α. Different logics are based on different definitions for entailment and consistency, and give us different options for argumentation. For a variety of logics, in particular for classical logic, the computational viability of generating arguments is an issue. Here, we present a solution that involves compiling a knowledgebase Δ based on the set of minimal inconsistent subsets of Δ, and then generating arguments from the compilation. Whilst generating a compilation is expensive, generating arguments from a compilation is relatively inexpensive.

Introduction

Argumentation is a vital aspect of intelligent behaviour by humans. Consider diverse professionals such as politicians, journalists, clinicians, scientists, and administrators, who all need to to collate and analyse information looking for pros and cons for consequences of importance when attempting to understand problems and make decisions.

There are a number of proposals for logic-based formalisations of argumentation (for reviews see (Prakken & Vreeswijk 2000; Chesnevar, Maguitman, & Loui 2001)). These proposals allow for the representation of arguments for and against some conclusion, and for attack relationships between arguments. In a number of key examples of argumentation systems, an argument is a pair where the first item in the pair is a minimal consistent set of formulae that proves the second item which is a formula (see for example (Benferhat, Dubois, & Prade 1993; Elvang-Gøransson, Krause, & Fox 1993; Besnard & Hunter 2001; Amgoud & Cayrol 2002; García & Simari 2004; Besnard & Hunter 2005)). Furthermore, in these approaches, the notion of attack is a form of undercut, where one argument undercuts another argument when the conclusion of the first argument negates the premises of the second argument.

In this paper, we consider viable algorithms for constructing undercuts. Let us assume that we construct undercuts incrementally: We start with an argument $\langle\Phi, \alpha\rangle$ and then we look for undercuts for this argument, and by recursion, we look for undercuts to undercuts. We wish to find all these arguments. This means a pruning strategy, such as incorporated into defeasible logic programming (García & Simari 2004; Chesnevar, Simari, & Godo 2005), would not meet our requirements since some undercuts would not be obtained.

Let us start by considering the construction of individual arguments. If Δ is a knowledgebase, and we are interested in a claim α, we seek an argument $\langle\Phi, \alpha\rangle$ where $\Phi \subseteq \Delta$. Deciding whether a set of propositional classical formulae is classically consistent is an NP-complete decision problem and deciding whether a set of propositional formulae classically entails a given formula is a co-NP-complete decision problem (Garey & Johnson 1979). However, if we consider the problem as an abduction problem, where we seek the existence of a minimal subset of a set of formulae that implies the consequent, then the problem is in the second level of the polynomial hierarchy (Eiter & Gottlob 1995). Even worse deciding whether a set of first-order classical formulae is consistent is an undecidable decision problem (Boolos, Burgess, & Jeffrey 2002). So even finding the basic units of argumentation is computationally challenging.

Proof procedures and algorithms have been developed for finding preferred arguments from a knowledgebase following for example Dung's preferred semantics (see for example (Prakken & Sartor 1997; Kakas & Toni 1999; Cayrol, Doutre, & Mengin 2001; Dimopoulos, Nebel, & Toni 2002; Dung, Kowalski, & Toni 2006)). However, these techniques and analyses do not offer any ways of ameliorating the computational complexity inherent in finding arguments and counterarguments even though it is a significant source of computational inefficiency.

A possible approach to ameliorate the cost of entailment is to use approximate entailment: Proposed in (Levesque 1984), and developed in (Schaerf & Cadoli 1995), classical entailment is approximated by two sequences of entailment relations. The first is sound but not complete, and the second is complete but not sound. Both sequences converge to classical entailment. For a set of propositional formulae Δ, a formula α, and an approximate entailment relation $\models_i$, the decision of whether $\Delta \models_i \alpha$ holds or $\Delta \not\models_i \alpha$ holds can be computed in polynomial time. Approximate entailment has been developed for anytime coherence reasoning (Koriche 2001; 2002). However, the approach still needs to be further developed and evaluated for finding arguments and counterarguments in argumentation. This would need to start with a conceptualization of the notions of argument and counteragument derived using approximate entailment and approximate coherence.

Another approximation approach is cost-bounded argumentation proposed for a form of probabilistic argumentation (Haenni 2001). For this, the cost is a function of the premises used for each argument, and this is justified in terms of probabilities associated with the formulae. In effect, the cost function biases which arguments are constructed in preference to others. This proposal may be adaptable for non-probabilistic argumentation, though given the biases that emanate from the choice of cost function, it is unlikely to be appropriate for all applications.

In this paper, we take a different approach by presenting a solution that involves compiling a knowledgebase, and then using the compilation to construct undercuts to an argument, and then construct undercuts to undercuts by recursion. The compilation of a knowledgebase Δ is based on the minimal inconsistent subsets of Δ.

We proceed as follows: We review an existing version of logic-based argumentation, which is the basis for our approach to compilation; We provide some propositions concerning minimal inconsistent subsets of a knowledgebase that we use to motivate the use of compilations; We present our definition for compilation together with a theorem relating compilations and argumentation; We present the algorithm for generating arguments from a compilation, and follow this with illustrative examples and completeness and correctness theorems; and We present an algorithm for generating a compilation from a knowledgebase.

Logical argumentation

In this section we review an existing proposal for logic-based argumentation (Besnard & Hunter 2001). We consider a classical propositional language. We use $\alpha, \beta, \gamma, \ldots$ to denote formulae and $\Delta, \Phi, \Psi, \ldots$ to denote sets of formulae. Deduction in classical propositional logic is denoted by the symbol $\vdash$ and deductive closure by Th so that $\mathsf{Th}(\Phi) = \{\alpha \mid \Phi \vdash \alpha\}$.

For the following definitions, we first assume a knowledgebase Δ (a finite set of formulae) and use this Δ throughout. We further assume that every subset of Δ is given an enumeration $\langle \alpha_1, \ldots, \alpha_n \rangle$ of its elements, which we call its canonical enumeration. This really is not a demanding constraint: In particular, the constraint is satisfied whenever we impose an arbitrary total ordering over Δ. Importantly, the order has no meaning and is not meant to represent any respective importance of formulae in Δ. It is only a convenient way to indicate the order in which we assume the formulae in any subset of Δ are conjoined to make a formula logically equivalent to that subset.

The paradigm for the approach is a large repository of information, represented by Δ, from which arguments can be constructed for and against arbitrary claims. Apart from information being understood as declarative statements, there is no a priori restriction on the contents, and the pieces of information in the repository can be as complex as possible. Therefore, Δ is not expected to be consistent. It need even not be the case that every single formula in Δ is consistent.

The framework adopts a very common intuitive notion of an argument. Essentially, an argument is a set of relevant formulae that can be used to classically prove some claim, together with that claim. Each claim is represented by a formula.

Definition 1. *An* **argument** *is a pair $\langle \Phi, \alpha \rangle$ such that: (1) $\Phi \subseteq \Delta$; (2) $\Phi \not\vdash \bot$; (3) $\Phi \vdash \alpha$; and (4) there is no $\Phi' \subset \Phi$ such that $\Phi' \vdash \alpha$. We say that $\langle \Phi, \alpha \rangle$ is an argument for α. We call α the* **claim** *(or consequent) of the argument and Φ the* **support** *of the argument (we also say that Φ is a support for α).*

Example 1. *Let $\Delta = \{\alpha, \alpha \rightarrow \beta, \gamma \rightarrow \neg\beta, \gamma, \delta, \delta \rightarrow \beta, \neg\alpha, \neg\gamma\}$. Some arguments are:*

$$\begin{gathered}\langle \{\alpha, \alpha \rightarrow \beta\}, \beta \rangle \\ \langle \{\neg\alpha\}, \neg\alpha \rangle \\ \langle \{\alpha \rightarrow \beta\}, \neg\alpha \vee \beta \rangle \\ \langle \{\neg\gamma\}, \delta \rightarrow \neg\gamma \rangle\end{gathered}$$

Arguments are not independent. In a sense, some encompass others (possibly up to some form of equivalence). To clarify this requires a few definitions as follows.

Definition 2. *An argument $\langle \Phi, \alpha \rangle$ is* **more conservative** *than an argument $\langle \Psi, \beta \rangle$ iff $\Phi \subseteq \Psi$ and $\beta \vdash \alpha$.*

Example 2. *$\langle \{\alpha\}, \alpha \vee \beta \rangle$ is more conservative than $\langle \{\alpha, \alpha \rightarrow \beta\}, \beta \rangle$.*

Some arguments directly oppose the support of others, which amounts to the notion of an undercut.

Definition 3. *An* **undercut** *for an argument $\langle \Phi, \alpha \rangle$ is an argument $\langle \Psi, \neg(\phi_1 \wedge \ldots \wedge \phi_n) \rangle$ where $\{\phi_1, \ldots, \phi_n\} \subseteq \Phi$.*

Example 3. *Let $\Delta = \{\alpha, \alpha \rightarrow \beta, \gamma, \gamma \rightarrow \neg\alpha\}$. Then, $\langle \{\gamma, \gamma \rightarrow \neg\alpha\}, \neg(\alpha \wedge (\alpha \rightarrow \beta)) \rangle$ is an undercut for $\langle \{\alpha, \alpha \rightarrow \beta\}, \beta \rangle$. A less conservative undercut for $\langle \{\alpha, \alpha \rightarrow \beta\}, \beta \rangle$ is $\langle \{\gamma, \gamma \rightarrow \neg\alpha\}, \neg\alpha \rangle$.*

Definition 4. *$\langle \Psi, \beta \rangle$ is a* **maximally conservative undercut** *of $\langle \Phi, \alpha \rangle$ iff $\langle \Psi, \beta \rangle$ is an undercut of $\langle \Phi, \alpha \rangle$ such that no undercuts of $\langle \Phi, \alpha \rangle$ are strictly more conservative than $\langle \Psi, \beta \rangle$ (that is, for all undercuts $\langle \Psi', \beta' \rangle$ of $\langle \Phi, \alpha \rangle$, if $\Psi' \subseteq \Psi$ and $\beta \vdash \beta'$ then $\Psi \subseteq \Psi'$ and $\beta' \vdash \beta$).*

The value of the following definition of canonical undercut is that we only need to take the canonical undercuts into account. This means we can justifiably ignore the potentially very large number of non-canonical undercuts.

Definition 5. *An argument $\langle \Psi, \neg(\phi_1 \wedge \ldots \wedge \phi_n) \rangle$ is a* **canonical undercut** *for $\langle \Phi, \alpha \rangle$ iff it is a maximally conservative undercut for $\langle \Phi, \alpha \rangle$ and $\langle \phi_1, \ldots, \phi_n \rangle$ is the canonical enumeration of Φ.*

An argument tree describes the various ways an argument can be challenged, as well as how the counter-arguments to the initial argument can themselves be challenged, and so on recursively.

Definition 6. *A* **complete argument tree** *for α is a tree where the nodes are arguments such that*

1. *The root is an argument for α.*
2. *For no node $\langle \Phi, \beta \rangle$ with ancestor nodes $\langle \Phi_1, \beta_1 \rangle, \ldots, \langle \Phi_n, \beta_n \rangle$ is Φ a subset of $\Phi_1 \cup \cdots \cup \Phi_n$.*
3. *The children nodes of a node N consist of all canonical undercuts for N that obey 2.*

The second condition in Definition 6 ensures that each argument on a branch has to introduce at least one formula in its support that has not already been used by ancestor arguments. This is meant to avoid making explicit undercuts that simply repeat over and over the same reasoning pattern except for switching the role of some formulae (e.g. in mutual exclusion, stating that α together with $\neg\alpha \vee \neg\beta$ entails $\neg\beta$ is exactly the same reasoning as expressing that β together with $\neg\alpha \vee \neg\beta$ entail $\neg\alpha$, because in both cases, what is meant is that α and β exclude each other).

As a notational convenience, in examples of argument trees the $\Diamond$ symbol is used to denote the consequent of an argument when that argument is a canonical undercut (no ambiguity arises as proven in (Besnard & Hunter 2001)).

Example 4. *Let $\Delta = \{\alpha \vee \beta, \alpha \to \gamma, \neg\gamma, \neg\beta, \delta \leftrightarrow \beta\}$. For this, two argument trees for the consequent $\alpha \vee \neg\delta$ are given.*

$$\begin{tikzcd}
\langle \{\alpha \vee \beta, \neg\beta\}, \alpha \vee \neg\delta \rangle & \langle \{\delta \leftrightarrow \beta, \neg\beta\}, \alpha \vee \neg\delta \rangle \\
\langle \{\alpha \to \gamma, \neg\gamma\}, \Diamond \rangle \arrow[u] & \langle \{\alpha \vee \beta, \alpha \to \gamma, \neg\gamma\}, \Diamond \rangle \arrow[u]
\end{tikzcd}$$

A complete argument tree is an efficient representation of the counterarguments, counter-counterarguments, ... Furthermore, if Δ is finite, there is a finite number of argument trees with the root being an argument with the claim α that can be formed from Δ, and each of these trees has finite branching and a finite depth (the finite tree property). Note, also the definitions presented in this section can be used directly with first-order classical logic, so Δ and α are from the first-order classical language. Interestingly, the finite tree property also holds for the first-order case (Besnard & Hunter 2005).

Minimal inconsistent subsets

If Δ is consistent, then there are no counterarguments, otherwise the minimal inconsistent subsets of a knowledgebase Δ summarise the conflicts that arise in Δ.

Definition 7. *For a knowledgebase Δ, the set of* **minimal inconsistent subsets** *of Δ, denoted $\mathsf{MinIncon}(\Delta)$, is defined as follows.*

$$\mathsf{MinIncon}(\Delta) = \{X \subseteq \Delta \mid X \vdash \bot \text{ and } \forall Y \subset X, Y \not\vdash \bot\}$$

We are interested in the minimal inconsistent subsets of Δ because of the following propositions.

Proposition 1. *If $\langle \Psi, \diamond \rangle$ is a canonical undercut, then there is an $X \in \mathsf{MinIncon}(\Delta)$, such that $\Psi \subset X$.*

Proof: Let $\langle \Phi, \alpha \rangle$ be the argument for which $\langle \Psi, \diamond \rangle$ is a canonical undercut. Therefore, $\Phi \cup \Psi \vdash \bot$. Therefore, there is a subset $\Phi' \subseteq \Phi$ where $\Phi' \cup \Psi \vdash \bot$ and for all $\Phi'' \subset \Phi'$, $\Phi'' \cup \Phi' \not\vdash \bot$. Therefore, $\Phi' \cup \Psi \in \mathsf{MinIncon}(\Delta)$ and $\Psi \subset (\Phi' \cup \Psi)$. □

However, if $\langle \Phi, \alpha \rangle$ is the root of an argument tree, it is not necessarily the case that there is an $X \in \mathsf{MinIncon}(\Delta)$, such that $\Phi \subset X$.

Example 5. *Let $\Delta = \{\alpha, \neg\alpha, \beta, \neg\beta\}$. Hence the minimal inconsistent subsets are $X_1 = \{\alpha, \neg\alpha\}$ and $X_2 = \{\beta, \neg\beta\}$. Now consider the argument at the root of an argument tree $\langle \Phi, \alpha \wedge \beta \rangle$ where $\Phi = \{\alpha, \beta\}$, $\Phi \not\subset X_1$ and $\Phi \not\subset X_2$.*

The following result shows how we can construct our canonical undercuts from the set of minimal inconsistent subsets of Δ.

Proposition 2. *Let $\langle \Phi, \alpha \rangle$ be an argument. For all $X \in \mathsf{MinIncon}(\Delta)$, if Φ is a non-empty subset of X, then $\langle X \setminus \Phi, \diamond \rangle$ is a canonical undercut for $\langle \Phi, \alpha \rangle$.*

Proof: Since $X \in \mathsf{MinIncon}(\Delta)$, and $\Phi \cap X \neq \emptyset$, we have $\Phi \cup (X \setminus \Phi) \vdash \bot$. Furthermore, there is no $W \subset (X \setminus \Phi)$ such that $\Phi \cup W \vdash \bot$. Therefore, $\langle X \setminus \Phi, \diamond \rangle$ is a canonical undercut of $\langle \Phi, \alpha \rangle$. □.

Example 6. *Let $\mathsf{MinIncon}(\Delta) = \{\{\beta, \neg\beta \vee \neg\gamma, \gamma\}\}$. Consider the argument $\langle \{\neg\beta \vee \neg\gamma\}, \neg(\beta \wedge \gamma) \rangle$. Clearly, $\{\neg\beta \vee \neg\gamma\}$ is a non-empty subset of $\{\beta, \neg\beta \vee \neg\gamma, \gamma\}$. Hence, $\langle \{\beta, \neg\beta \vee \neg\gamma, \gamma\} \setminus \{\neg\beta \vee \neg\gamma\}, \diamond \rangle$ is a canonical undercut of $\langle \{\neg\beta \vee \neg\gamma\}, \neg(\beta \wedge \gamma) \rangle$.*

We generalize Proposition 2 as follows.

Proposition 3. *Let $\langle \Phi, \alpha \rangle$ be an argument. For all $X \in \mathsf{MinIncon}(\Delta)$, if $\Phi \cap X \neq \emptyset$, and there is no $Y \in \mathsf{MinIncon}(\Delta)$, such that $(Y \setminus \Phi) \subset (X \setminus \Phi)$, then $\langle X \setminus \Phi, \diamond \rangle$ is a canonical undercut for $\langle \Phi, \alpha \rangle$.*

Example 7. *Let $\mathsf{MinIncon}(\Delta) = \{X_1, X_2\}$ where $X_1 = \{\alpha, \neg\alpha\}$ and $X_2 = \{\beta, \neg\beta\}$. Now consider the argument $\langle \Phi, \alpha \wedge \beta \rangle$ where $\Phi = \{\alpha, \beta\}$. Here $\langle X_1 \setminus \Phi, \diamond \rangle$ and $\langle X_2 \setminus \Phi, \diamond \rangle$ are each a canonical undercut for $\langle \Phi, \alpha \rangle$.*

We get the converse of Proposition 2 as follows.

Proposition 4. *If $\langle \Psi, \diamond \rangle$ is a canonical undercut for $\langle \Phi, \alpha \rangle$, then there is an X such that $X \in \mathsf{MinIncon}(\Delta)$ and $\Psi = X \setminus \Phi$.*

Proof: Assume $\langle \Psi, \diamond \rangle$ is canonical undercut for $\langle \Phi, \alpha \rangle$. Therefore, $\Psi \vdash \neg(\wedge\Phi)$ and there is no $\Psi' \subset \Psi$ such that $\Psi' \vdash \neg(\wedge\Phi)$. Therefore, $\Psi \cup \Phi \vdash \bot$ and there is no $\Psi' \subset \Psi$ such that $\Psi' \cup \Phi \vdash \bot$. Therefore, there is a $\Phi' \subseteq \Phi$ such that $\Psi \cup \Phi' \in \mathsf{MinIncon}(\Delta)$. Let X be $\Psi \cup \Phi'$. Furthermore, $(X \setminus \Phi) \cup \Phi \vdash \bot$ and there is no $Z \subset (X \setminus \Phi)$ such that $Z \cup \Phi \vdash \bot$. Hence, $\Psi = X \setminus \Phi$. □

Example 8. *Let $\Delta = \{\beta, \beta \to \alpha, \beta \to \neg\alpha\}$. There is only one minimal inconsistent subset which is $\{\beta, \beta \to \alpha, \beta \to \neg\alpha\}$. Now consider the argument $\langle \{\beta, \beta \to \alpha\}, \alpha \rangle$. There is only one canonical undercut which is $\langle \{\beta \to \neg\alpha\}, \diamond \rangle$. Furthermore, we have the following holding between the support of the argument, the support of the undercut, and the minimal inconsistent subset.*

$$\{\beta \to \neg\alpha\} = \{\beta, \beta \to \alpha, \beta \to \neg\alpha\} \setminus \{\beta, \beta \to \alpha\}$$

Minimal inconsistent subsets of a knowledgebase are not necessarily disjoint. Consider $X, Y \in \mathsf{MinIncon}(\Delta)$ where $X \cap Y \neq \emptyset$. Furthermore, if the support of an argument $\langle \Phi, \alpha \rangle$ is a subset of X, and $Y \cap \Phi$ is non-empty, and there is no $Z \in \mathsf{MinIncon}(\Delta)$, such that $(Z \setminus \Phi) \subset (Y \setminus \Phi)$, then $\langle Y \setminus \Phi, \diamond \rangle$ is a canonical undercut for $\langle \Phi, \alpha \rangle$, as captured in the following result.

Proposition 5. *Let $\langle \Phi, \alpha \rangle$ be an argument. For all $X, Y \in \mathsf{MinIncon}(\Delta)$, if $X \cap Y \neq \emptyset$, and $\Phi \subseteq X$, and $Y \cap \Phi \neq \emptyset$, and there is no $Z \in \mathsf{MinIncon}(\Delta)$, such that $(Z \setminus \Phi) \subset (Y \setminus \Phi)$, then $\langle Y \setminus \Phi, \diamond \rangle$ is a canonical undercut for $\langle \Phi, \alpha \rangle$.*

Proof: Let $\langle \Phi, \alpha \rangle$ be an argument, and let $X, Y \in \mathsf{MinIncon}(\Delta)$, be such that $X \cap Y \neq \emptyset$, and $\Phi \subseteq X$, and $Y \cap \Phi \neq \emptyset$. Therefore $(Y \setminus \Phi) \cup \Phi \vdash \bot$. Also assume there is no $Z \in \mathsf{MinIncon}(\Delta)$ such that $(Z \setminus \Phi) \subset (Y \setminus \Phi)$. Therefore there is no $W \subset (Y \setminus \Phi)$ such that $W \cup \Phi \vdash \bot$. Therefore $\langle Y \setminus \Phi, \diamond \rangle$ is a canonical undercut for $\langle \Phi, \alpha \rangle$. $\Box$

Example 9. *Let $X_1, X_2 \subset \Delta$ where $X_1 = \{\alpha, \alpha \rightarrow \neg\beta, \beta\}$ and $X_2 = \{\alpha, \neg\alpha\}$. So $X_1, X_2 \in \mathsf{MinIncon}(\Delta)$ and $X_1 \cap X_2 \neq \emptyset$. Now consider the argument $\langle \Phi, \neg\beta \rangle$ where $\Phi = \{\alpha, \alpha \rightarrow \neg\beta\}$. So $\Phi \subset X_1$ and $\Phi \cap X_2 \neq \emptyset$. Furthermore, $\langle X_2 \setminus \Phi, \diamond \rangle$ is an undercut for $\langle \Phi, \diamond \rangle$.*

Example 10. *Let $X_1, X_2, X_3 \subset \Delta$ where*

$$\begin{gathered} X_1 = \{\neg\alpha, \neg\beta, \alpha \vee \beta\} \\ X_2 = \{\neg\alpha, \alpha \vee \neg\beta \vee \neg\gamma, \gamma, \beta\} \\ X_3 = \{\beta, \neg\beta\} \end{gathered}$$

Let $\mathsf{MinIncon}(\Delta) = \{\{X_1, X_2, X_3\}\}$. Now consider the argument $\langle \{\alpha \vee \beta\}, \alpha \vee \beta \rangle$. Let $\Phi = X_1 \setminus \{\alpha \vee \beta\} = \{\neg\alpha, \neg\beta\}$. So $\langle \Phi, \diamond \rangle$ is a canonical undercut for $\langle \{\alpha \vee \beta\}, \alpha \vee \beta \rangle$. Furthermore, $\Phi \cap X_2 \neq \emptyset$ and $\Phi \cap X_3 \neq \emptyset$. However, $\langle X_2 \setminus \Phi, \diamond \rangle$ is not a canonical undercut for $\langle \Phi, \diamond \rangle$, because $(X_3 \setminus \Phi) \subset (X_2 \setminus \Phi)$, whereas $\langle X_3 \setminus \Phi, \diamond \rangle$ is a canonical undercut for $\langle \Phi, \diamond \rangle$.

We also see that every canonical undercut for a canonical undercut can be found using these overlapping minimal inconsistent subsets of a knowledgebase.

Proposition 6. *Let $\langle \Phi, \diamond \rangle$ be a canonical undercut. If $\langle \Psi, \diamond \rangle$ is a canonical undercut of $\langle \Phi, \diamond \rangle$ then $\Psi \subset Y$ for some $Y \in \mathsf{MinIncon}(\Delta)$ such that $X \cap Y \neq \emptyset$ for every $X \in \mathsf{MinIncon}(\Delta)$ satisfying $\Phi \subset X$.*

Proof: Suppose $\langle \Psi, \diamond \rangle$ is a canonical undercut of $\langle \Phi, \diamond \rangle$. Therefore, $\Psi \cup \Phi \vdash \bot$ and there is not a $\Psi' \subset \Psi$ such that $\Psi' \cup \Phi \vdash \bot$. Therefore, there is a $Y \in \mathsf{MinIncon}(\Delta)$ such that $\Psi \subset Y$ and $\Phi \cap Y \neq \emptyset$. Therefore, for every $X \in \mathsf{MinIncon}(\Delta)$ such that $\Phi \subset X$, it immediately follows that $X \cap Y \neq \emptyset$. $\Box$

These results indicate that for any knowledgebase Δ, if we have $\mathsf{MinIncon}(\Delta)$, we can find all undercuts to any argument without recourse to a consequence relation or a satisfiability check.

Compilation of a knowledgebase

Argument compilation is a process that involves finding all the minimal inconsistent subsets of Δ, and then forming a hypergraph from this collection of subsets. Given a knowledgebase Δ, the function $\mathsf{Compilation}(\Delta)$ gives the hypergraph that can then be used for efficiently building argument trees.

Definition 8. *Let $\mathsf{Compilation}(\Delta) = (N, A)$ where $N = \mathsf{MinIncon}(\Delta)$ and*

$$A = \{(X, Y) \mid X, Y \in N \text{ and } X \cap Y \neq \emptyset\}$$

A graph is bidirectional if for each arc (X, Y) in the graph, (Y, X) is also in the graph. Clearly, for all Δ, $\mathsf{Compilation}(\Delta)$ is a bidirectional hypergraph. Though $\mathsf{Compilation}(\Delta)$ may be disjoint. Consider for example, $\Delta = \{\alpha, \neg\alpha, \beta, \neg\beta\}$.

Since the hypergraphs are bidirectional, we will represent them in the examples by just giving one of (X, Y) and (Y, X) for each of the arcs.

Example 11. *Let $\Delta = \{\delta \wedge \neg\alpha, \alpha, \neg\alpha, \alpha \vee \beta, \neg\beta\}$. Hence $\mathsf{MinIncon}(\Delta) = \{\ X_1,\ X_2,\ X_3,\ X_4\ \}$, where $X_1 = \{\delta \wedge \neg\alpha, \alpha\}$, $X_2 = \{\alpha, \neg\alpha\}$, $X_3 = \{\neg\alpha, \alpha \vee \beta, \neg\beta\}$, and $X_4 = \{\delta \wedge \neg\alpha, \alpha \vee \beta, \neg\beta\}$. So the nodes and arcs of the hypergraph are such that $N = \{X_1, X_2, X_3, X_4\}$ and $A = \{\ (X_1, X_2), (X_2, X_3), (X_3, X_4), (X_1, X_4)\ \}$.*

The following result shows that an argument tree is composed from subgraphs of the compilation.

Theorem 1. *Let $\mathsf{Compilation}(\Delta) = (N, A)$. If $\langle \Phi, \alpha \rangle$ is the root of a complete argument tree T and $\langle \Phi, \alpha \rangle$ is the parent node of $\langle \Psi, \diamond \rangle$ then the subtree of T rooted at $\langle \Psi, \diamond \rangle$ is isomorphic to a subgraph of (N, A).*

Proof: Consider a branch $\langle \Psi_0, \alpha \rangle, \langle \Psi_1, \diamond \rangle, \ldots, \langle \Psi_m, \diamond \rangle$ in T (i.e., Ψ_0 is Φ). The main step is to show that for $1 \leq i \leq m-1$, the fact that $\langle \Psi_{i+1}, \diamond \rangle$ is a canonical undercut for $\langle \Psi_i, \diamond \rangle$ entails that there exist X_i and X_{i+1} in N satisfying $(X_i, X_{i+1}) \in A$.
Applying Proposition 4, there exists $X_i \in N$ such that $\Psi_i = X_i \setminus \Psi_{i-1}$ and there exists $X_{i+1} \in N$ such that $\Psi_{i+1} = X_{i+1} \setminus \Psi_i$. Clearly, $X_i = \Psi_i \cup \Psi_{i-1}$ and $X_{i+1} = \Psi_{i+1} \cup \Psi_i$. Since Ψ_i is trivially disjoint from Ψ_{i-1} and Ψ_{i+1}, it then follows that $X_i = X_{i+1}$ implies $\Psi_{i-1} = \Psi_{i+1}$. By the definition of an argument tree, the support of an argument in T cannot be a subset of the support of an ancestor: Here, $\Psi_{i-1} \neq \Psi_{i+1}$. Therefore, $X_i \neq X_{i+1}$. As $X_i \cap X_{i+1} \neq \emptyset$ is trivial, $(X_i, X_{i+1}) \in A$ ensues.
There remains to show that no such branch is accounted for by an unfolding of a cycle in (N, A). Assume $X_j = X_{j+k}$ for some j and k. Similarly to the above observation, $X_j = \Psi_j \cup \Psi_{j-1}$ and $X_{j+k} = \Psi_{j+k} \cup \Psi_{j+k-1}$. So, $\Psi_j \cup \Psi_{j-1} = \Psi_{j+k} \cup \Psi_{j+k-1}$. Hence, $\Psi_{j+k} \subseteq \Psi_j \cup \Psi_{j-1}$. However, the definition of an argument tree prohibits the support of an argument to be a subset of the set-theoretic union of the supports of its ancestors. That is, a contradiction arises. Therefore, $X_j = X_{j+k}$ is impossible.
Lastly, it is immediate to notice that two siblings nodes cannot be accounted for by the same element in (N, A). Indeed, consider $\langle \Psi', \diamond \rangle$ and $\langle \Psi'', \diamond \rangle$ in T sharing the same parent node $\langle \Psi, \diamond \rangle$. Assume that Proposition 4 yields the same X in N for $\langle \Psi', \diamond \rangle$ and $\langle \Psi'', \diamond \rangle$. Then, $\Psi' = X \setminus \Psi$ and $\Psi'' = X \setminus \Psi$. Trivially, $\Psi' = \Psi''$ ensues but no two

distinct canonical undercuts for the same argument can have the same support. Therefore, the assumption must be false and the proof is over. □

Corollary 1. *Let* Compilation$(\Delta) = (N, A)$. *If* $\langle\Psi, \diamond\rangle$ *is canonical undercut for* $\langle\Phi, \diamond\rangle$, *and* $\langle\Phi, \diamond\rangle$ *is itself a canonical undercut, then there is an* $(X, Y) \in A$, *such that* $\Phi \subset X$ *and* $\Psi \subset Y$.

So by walking over Compilation(Δ), we can construct argument trees from Δ. If we have an argument for the root of the argument tree, then we can find undercuts and by recursion, undercuts to undercuts, by following the arcs of the compilation. Moreover, we will argue in the next section, the cost of using the compilation is much lower than constructing arguments directly from Δ.

Generating arguments from a compilation

Whilst argument compilation is expensive, once Compilation(Δ) has been computed, it can be used relatively cheaply. Assume that we want to construct a complete argument tree from Δ with the argument $\langle\Phi, \alpha\rangle$ as its root. Now we want to use the support of this argument together with the hypergraph (i.e. Compilation(Δ)) to systematically build the argument tree.

The way we proceed is to find which nodes in the hypergraph have a non-empty set intersection with Φ. Suppose, X is a node in the hypergraph and $\Phi \cap X \neq \emptyset$, then $X \setminus \Phi$ is the support of a counterargument to $\langle\Phi, \alpha\rangle$. Indeed, if there is no node Y such that $\Phi \cap Y \neq \emptyset$ and $(Y \setminus \Phi) \subset (X \setminus \Phi)$, then $\langle X \setminus \Phi, \diamond\rangle$ is a canonical undercut of $\langle\Phi, \alpha\rangle$ (according to Proposition 3).

We now proceed by recursion looking for canonical undercuts to the canonical undercuts. For this we use the arcs of the hypergraph. Suppose, $\langle\Gamma_j, \diamond\rangle$ is a canonical undercut in the tree. Let $\langle\Gamma_i, \diamond\rangle$ be the parent of $\langle\Gamma_j, \diamond\rangle$ (i.e $\langle\Gamma_j, \diamond\rangle$ is a canonical undercut of $\langle\Gamma_i, \diamond\rangle$). From this, we know there is a node in the hypergraph X such that $\Gamma_j = X \setminus \Gamma_i$ (c.f. Proposition 4). Hence, $\Gamma_j \subset X$. Moreover, if there is an arc in the hypergraph (X, Y) and $Y \cap \Gamma_i \neq \emptyset$, and there is no arc (X, Z) in the hypergraph such that $Z \setminus \Phi \subset Y \setminus \Phi$, then $\langle Y \setminus \Gamma_j, \diamond\rangle$ is a canonical undercut to $\langle\Gamma_j, \diamond\rangle$ (according to Proposition 5).

Proposition 6 justifies that we pick *one* X, such that X is a superset of Φ, and then we can find all the other canonical undercuts of $\langle\Phi, \diamond\rangle$ by just following the arcs in Compilation(Δ).

By keeping track of which arcs we have traversed in the hypergraph, we can efficiently build a complete argument tree. The `Undercuts` algorithm, presented in Definition 9, assumes that the argument for the root of the argument tree is given as input. Then the algorithm determines where this root argument is located in the hypergraph, and undertakes a depth-first search identifying canonical undercuts by recursion. So for a knowledgebase Δ, the `Undercuts` algorithm, together with subsidiary algorithms, only use Compilation(Δ).

Definition 9. *The algorithm* `Undercuts` *takes the root of an argument tree* $\langle\Phi, \alpha\rangle$ *and the hypergraph* Compilation$(\Delta) = (N, A)$, *and returns the nodes of the argument tree. The first* add *statement puts the undercuts to the root into the solution, and the second* add *statement uses the algorithm* `Subcuts` *to find undercuts to undercuts by recursion and puts them into the solution.*

```
Undercuts($\langle\Phi, \alpha\rangle, N, A$)
    FirstLevelSets = FirstLevel($\langle\Phi, \alpha\rangle, N, A$)
    Output = $\emptyset$
    while FirstLevelSets $\neq \emptyset$
        remove $X$ from FirstLevelSets
        add $\langle X \setminus \Phi, \diamond\rangle$ is an undercut of $\langle\Phi, \alpha\rangle$ to Output
        add Subcuts($X \setminus \Phi, N \setminus \{X\}, A, X, \Phi$) to Output
    return Output
```

The algorithm `FirstLevel` *takes the root of an argument tree* $\langle\Phi, \alpha\rangle$ *and the hypergraph* Compilation$(\Delta) = (N, A)$, *and returns the subset of* N *that can be used for constructing the undercuts to the root.*

```
FirstLevel($\langle\Phi, \alpha\rangle, N, A$)
    Candidates = $\{X \in N \mid \Phi \cap X \neq \emptyset\}$
    Output = $\emptyset$
    while Candidates $\neq \emptyset$
        remove $X$ from Candidates
        if there is no $Y \in$ Candidates $\cup$ Output
                s.t. $(Y \setminus \Phi) \subset (X \setminus \Phi)$
        then add $X$ to Output
    return Output
```

The algorithm `Subcuts` *is a depth-first search algorithm that takes the support of an argument,* Γ, *the set of available hypernodes* M *in the depth-first search, the current hypernode* X, *and the current union of the supports for the arguments used on the branch up to the root* S.

```
Subcuts($\Gamma, M, A, X, S$)
    Output = $\emptyset$
    for all $Y \in M$
        if $(X,Y) \in A$ and $Y \not\subseteq S$ and $Y \cap \Gamma \neq \emptyset$
                and there is no $(X, Z) \in A$
                        such that $(Z \setminus \Gamma) \subset (Y \setminus \Gamma)$
        then add $\langle Y \setminus \Gamma, \diamond\rangle$ is an undercut of $\langle\Gamma, \diamond\rangle$
                                        to Output
        and add Subcuts($Y \setminus \Gamma, M \setminus \{Y\}, A, Y, S \cup Y$)
                                        to Output
    return Output
```

We illustrate the key features of the `Undercuts` algorithm in Example 12 which shows how the `FirstLevel` algorithm finds the nodes in N that intersect the root of the support, how these are then used to generate the undercuts to the root, and how the `Subcuts` algorithm finds the undercuts to the undercuts.

Note, in the `FirstLevel` algorithm, that for each candidate node X, there is the following check (on the 6th line of the algorithm).

$$\text{there is no } Y \in \texttt{Candidates} \cup \texttt{Output} \text{ such that } (Y \setminus \Phi) \subset (X \setminus \Phi)$$

This is to ensure that the support for each undercut is minimal: The support of the root can involve formulae from more than one minimally inconsistent set of formulae and so it is not the case that every node X with a non-empty intersection with Φ is such that $X \setminus \Phi$ is minimal (as illustrated in Example 13).

Note, in `Subcuts` we check that $Y \cap \Gamma \neq \emptyset$ because we want to ensure that $Y \setminus \Gamma$ is inconsistent with Γ rather than with the support of the parent of $\langle \Gamma, \diamond \rangle$ (which is illustrated in Example 14). Also, in `Subcuts`, we check that $Y \not\subseteq S$, because according to the definition of an argument tree, no undercut on a branch can have a support that is a subset of the union of the support of the ancestors.

Example 12. *Consider* $\Delta = \{\alpha, \neg\alpha, \beta \wedge \neg\beta, \gamma, \gamma \rightarrow \delta, \neg\delta, \neg\gamma \vee \delta\}$ *which has the following compilation* (N, A)*, where* $N = \{X_1, X_2, X_3, X_4\}$ *and* $A = \{(X_4, X_3)\}$*.*

$$\begin{aligned} X_1 &= \{\alpha, \neg\alpha\} \\ X_2 &= \{\beta \wedge \neg\beta\} \\ X_3 &= \{\gamma, \gamma \rightarrow \delta, \neg\delta\} \\ X_4 &= \{\gamma, \neg\gamma \vee \delta, \neg\delta\} \end{aligned}$$

Now suppose, we have the following argument as the root of the argument tree.

$$\langle \{\gamma, \gamma \rightarrow \delta\}, \delta \vee \beta \rangle$$

We have support for one canonical undercut of the root (i.e. `FirstLevelSets` $= \{X_3\}$*).*

$$X_3 \setminus \{\gamma, \gamma \rightarrow \delta\} = \{\neg\delta\}$$

Also we have (X_4, X_3) *as an arc and so we have an undercut to the undercut.*

$$X_4 \setminus \{\neg\delta\} = \{\gamma, \neg\gamma \vee \delta\}$$

There are no more arcs to consider and so the net result is the following argument tree. The elements of N *that are used are given adjacent to the arrow in the following diagram.*

$$\begin{tikzcd}
\langle \{\gamma, \gamma \rightarrow \delta\}, \delta \vee \beta \rangle \\
\langle \{\neg\delta\}, \diamond \rangle \arrow[u, "X_3"'] \\
\langle \{\gamma, \neg\gamma \vee \delta\}, \diamond \rangle \arrow[u, "X_4"']
\end{tikzcd}$$

In this example, we see that not all nodes are used in the construction of this tree. Furthermore, one node will never be used in any argument tree (i.e. X_2*).*

Now suppose we have the following argument as the root of the argument tree.

$$\langle \{\gamma, \gamma \rightarrow \delta, \alpha\}, \delta \wedge \alpha \rangle$$

We have supports for two canonical undercuts of the root (i.e. `FirstLevelSets` $= \{X_1, X_3\}$*).*

$$\begin{aligned} X_1 \setminus \{\gamma, \gamma \rightarrow \delta, \alpha\} &= \{\neg\alpha\} \\ X_3 \setminus \{\gamma, \gamma \rightarrow \delta, \alpha\} &= \{\neg\delta\} \end{aligned}$$

Also we have (X_4, X_3) *as an arc and so we have an undercut to the second of the undercuts.*

$$X_4 \setminus \{\neg\delta\} = \{\gamma, \neg\gamma \vee \delta\}$$

There are no more arcs to consider and so the net result is the following argument tree.

$$\begin{tikzcd}
& \langle \{\gamma, \gamma \rightarrow \delta, \alpha\}, \delta \wedge \alpha \rangle & \\
\langle \{\neg\alpha\}, \diamond \rangle \arrow[ur, "X_1"] & & \langle \{\neg\delta\}, \diamond \rangle \arrow[ul, "X_3"'] \\
& & \langle \{\gamma, \neg\gamma \vee \delta\}, \diamond \rangle \arrow[u, "X_4"']
\end{tikzcd}$$

Example 13. *Let* $\Delta = \{\alpha, \beta, \neg\beta, \neg\alpha \vee \beta\}$*. So* $N = \{X_1, X_2\}$*, where* $(X_1, X_2) \in A$*, for* X_1 *and* X_2 *as follows.*

$$\begin{aligned} X_1 &= \{\alpha, \neg\beta, \neg\alpha \vee \beta\} \\ X_2 &= \{\beta, \neg\beta\} \end{aligned}$$

Now consider the root of an argument tree $\langle \Phi, \alpha \wedge \beta \rangle$*, where* $\Phi = \{\alpha, \beta\}$*. Here we have* $X_1 \cap \Phi \neq \emptyset$ *and* $X_2 \cap \Phi \neq \emptyset$*. So both are candidates for forming supports for undercuts for the root argument.*

$$\begin{aligned} X_1 \setminus \Phi &= \{\neg\beta, \neg\alpha \vee \beta\} \\ X_2 \setminus \Phi &= \{\neg\beta\} \end{aligned}$$

However, from the above, we see that we have the following.

$$X_2 \setminus \Phi \subseteq X_1 \setminus \Phi$$

Hence, X_1 *fails the condition on line 6 of the* `FirstLevel` *algorithm, whereas* X_2 *succeeds. Therefore, only* X_2 *is used to construct an undercut to the root. Also we have* (X_1, X_2) *as an arc and so we have an undercut to the undercut. The net result is the following argument tree.*

$$\begin{tikzcd}
\langle \{\alpha, \beta\}, \alpha \wedge \beta \rangle \\
\langle \{\neg\beta\}, \diamond \rangle \arrow[u] \\
\langle \{\alpha, \neg\alpha \vee \beta\}, \diamond \rangle \arrow[u]
\end{tikzcd}$$

Example 14. *Let* $\Delta = \{\gamma \wedge \alpha, \beta, \beta \rightarrow \neg\alpha, \neg\gamma\}$*. Hence,* $N = \{X_1, X_2\}$*, where* $(X_1, X_2) \in A$*, for* X_1 *and* X_2 *as follows.*

$$\begin{aligned} X_1 &= \{\gamma \wedge \alpha, \beta, \beta \rightarrow \neg\alpha\} \\ X_2 &= \{\gamma \wedge \alpha, \neg\gamma\} \end{aligned}$$

For this, we get the following argument tree with root $\langle \{\gamma \wedge \alpha\}, \alpha \rangle$*.*

$$\begin{tikzcd}
& \langle \{\gamma \wedge \alpha\}, \alpha \rangle & \\
\langle \{\beta, \beta \rightarrow \neg\alpha\}, \diamond \rangle \arrow[ur, "X_1"] & & \langle \{\neg\gamma\}, \diamond \rangle \arrow[ul, "X_2"']
\end{tikzcd}$$

Note, even though we have (X_1, X_2) *in* A*, we do not get an undercut for* $\langle \{\beta, \beta \rightarrow \neg\alpha\}, \diamond \rangle$ *using* X_2 *because of the following which fails the condition on line 3 of the* `Subcuts` *algorithm.*

$$\{\beta, \beta \rightarrow \neg\alpha\} \cap X_2 = \emptyset$$

For the same reason, we do not get an undercut for $\langle \{\neg\gamma\}, \diamond \rangle$ *using* X_1*.*

Example 15. *As another illustration, continuing Example 11, suppose* $\langle \{\alpha \vee \beta\}, \alpha \vee \beta \rangle$ *is the root of the tree. We have supports for two canonical undercuts of the root which are*

$$\begin{aligned} X_3 \setminus \{\alpha \vee \beta\} &= \{\neg\alpha, \neg\beta\} \\ X_4 \setminus \{\alpha \vee \beta\} &= \{\delta \wedge \neg\alpha, \neg\beta\} \end{aligned}$$

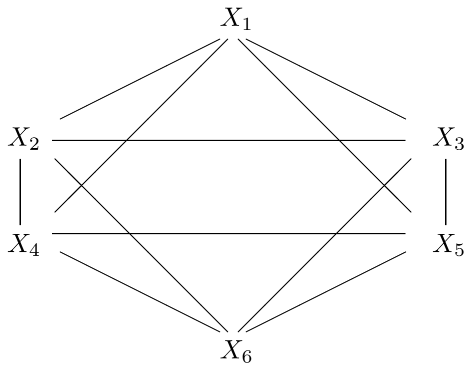

Figure 1: The compilation for Example 16 with nodes X_1,..,X_6.

We have (X_3, X_2), (X_3, X_4), and (X_4, X_1) as arcs, and so we have undercuts to undercuts as following.

$$\begin{array}{l} X_2 \setminus \{\neg\alpha, \neg\beta\} = \{\alpha\} \\ X_4 \setminus \{\neg\alpha, \neg\beta\} = \{\delta \wedge \neg\alpha, \alpha \vee \beta\} \\ X_1 \setminus \{\delta \wedge \neg\alpha, \neg\beta\} = \{\alpha\} \\ X_3 \setminus \{\delta \wedge \neg\alpha, \neg\beta\} = \{\neg\alpha, \alpha \vee \beta\} \end{array}$$

Finally, for the above undercuts, we have (X_4, X_1), (X_2, X_1), and (X_3, X_2) as arcs, and so we have undercuts to undercuts as following.

$$\begin{array}{l} X_1 \setminus \{\delta \wedge \neg\alpha, \alpha \vee \beta\} = \{\alpha\} \\ X_1 \setminus \{\alpha\} = \{\delta \wedge \neg\alpha\} \\ X_2 \setminus \{\alpha\} = \{\neg\alpha\} \\ X_2 \setminus \{\neg\alpha, \alpha \vee \beta\} = \{\alpha\} \end{array}$$

The net result is the following argument tree labelled with the element from N used for finding each undercut.

$$\begin{tikzcd}
 & & & \langle\{\alpha \vee \beta\}, \alpha \vee \beta\rangle & & & \\
 & \langle\{\neg\alpha, \neg\beta\}, \diamond\rangle \arrow[urr, "X_3"] & & & & \langle\{\delta \wedge \neg\alpha, \neg\beta\}, \diamond\rangle \arrow[ull, "X_4"'] & \\
\langle\{\delta \wedge \neg\alpha, \alpha \vee \beta\}, \diamond\rangle \arrow[ur, "X_4"] & & \langle\{\alpha\}, \diamond\rangle \arrow[ul, "X_2"'] & & \langle\{\alpha\}, \diamond\rangle \arrow[ur, "X_1"] & & \langle\{\neg\alpha, \alpha \vee \beta\}, \diamond\rangle \arrow[ul, "X_3"'] \\
\langle\{\alpha\}, \diamond\rangle \arrow[u, "X_1"] & & \langle\{\delta \wedge \neg\alpha\}, \diamond\rangle \arrow[u, "X_1"] & & \langle\{\neg\alpha\}, \diamond\rangle \arrow[u, "X_2"] & & \langle\{\alpha\}, \diamond\rangle \arrow[u, "X_2"]
\end{tikzcd}$$

Example 16. *This example represents an extreme case of inconsistency in a knowledgebase. Let $\Delta = \{\alpha \wedge \beta, \alpha \wedge \neg\beta, \neg\alpha \wedge \beta, \neg\alpha \wedge \neg\beta\}$, let $N = \{ X_1, X_2, X_3, X_4, X_5, X_6 \}$, and let $A = \{(X_1, X_2), (X_1, X_3), (X_1, X_4), (X_1, X_5), (X_2, X_3), (X_2, X_4), (X_2, X_6), (X_3, X_5), (X_3, X_6), (X_4, X_5), (X_4, X_6), (X_5, X_6)\}$, which is illustrated in Figure 1, and where*

$$\begin{array}{ll} X_1 = \{\alpha \wedge \beta, \alpha \wedge \neg\beta\} & X_2 = \{\alpha \wedge \beta, \neg\alpha \wedge \beta\} \\ X_3 = \{\alpha \wedge \beta, \neg\alpha \wedge \neg\beta\} & X_4 = \{\alpha \wedge \neg\beta, \neg\alpha \wedge \beta\} \\ X_5 = \{\alpha \wedge \neg\beta, \neg\alpha \wedge \neg\beta\} & X_6 = \{\neg\alpha \wedge \beta, \neg\alpha \wedge \neg\beta\} \end{array}$$

Let

$$\begin{array}{l} A_1 = \langle\{\alpha \wedge \beta\}, \alpha \wedge \beta\rangle \\ A_2 = \langle\{\alpha \wedge \neg\beta\}, \alpha \wedge \neg\beta\rangle \\ A_3 = \langle\{\neg\alpha \wedge \beta\}, \neg\alpha \wedge \beta\rangle \\ A_4 = \langle\{\neg\alpha \wedge \neg\beta\}, \neg\alpha \wedge \neg\beta\rangle \end{array}$$

Hence, we get the following argument tree. Each arc is labelled with the index i of the set $X_i \in N$ used to construct the undercut.

$$\begin{tikzcd}
 & & & & & A_1 & & & & & \\
 & A_2 \arrow[urrrr, "1"] & & & & A_3 \arrow[u, "2"] & & & & A_4 \arrow[ullll, "3"'] & \\
A_3 \arrow[ur, "4"] & & A_4 \arrow[ul, "5"'] & & A_2 \arrow[ur, "4"] & & A_4 \arrow[ul, "6"'] & & A_2 \arrow[ur, "5"] & & A_3 \arrow[ul, "6"'] \\
A_4 \arrow[u, "6"] & & A_3 \arrow[u, "6"] & & A_4 \arrow[u, "5"] & & A_2 \arrow[u, "5"] & & A_3 \arrow[u, "4"] & & A_2 \arrow[u, "4"]
\end{tikzcd}$$

Assuming some knowledgebase Δ, the following completeness result shows that for any argument, we get all the canonical undercuts for it using the compilation of Δ.

Theorem 2. *For a knowledgebase Δ, let $\mathsf{Compilation}(\Delta) = (N, A)$, Let $\langle\Gamma, \alpha\rangle$ be an argument. For all $X \in N$ such that $\Gamma \subseteq X$, if $\langle\Psi, \diamond\rangle$ is an undercut for $\langle\Gamma, \alpha\rangle$ then*

- *either $\Psi = X \setminus \Gamma$*
- *or $\Psi = Y \setminus \Gamma$ for some $Y \in N$ such that $(X, Y) \in A$ and $Y \cap \Gamma \neq \emptyset$*

Proof: Let $X \in N$ satisfy $\Gamma \subseteq X$ and let $\langle\Psi, \diamond\rangle$ be an undercut for $\langle\Gamma, \alpha\rangle$. It is sufficient to make the hypothesis $\Psi \neq X \setminus \Gamma$ and then prove $\Psi = Y \setminus \Gamma$ for some $Y \in N$ such that $Y \cap \Gamma \neq \emptyset$ and $(X, Y) \in A$.
$\langle\Psi, \diamond\rangle$ is an undercut for $\langle\Gamma, \alpha\rangle$, hence Ψ is a minimal consistent subset of Δ such that $\Psi \vdash \neg(\gamma_1 \wedge \ldots \wedge \gamma_n)$ where $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$. By classical logic, there exists a minimal subset $\Gamma' = \{\gamma'_1, \ldots, \gamma'_m\}$ of Γ that satisfies $\Psi \vdash \neg(\gamma'_1 \wedge \ldots \wedge \gamma'_m)$ for some $m > 0$ (i.e., Γ' is non-empty). So, $\Psi \cup \Gamma'$ is minimally inconsistent.
Reasoning by reductio ad absurdum, assume some $\gamma_i \in \Gamma \cap \Psi$. Then, $\langle\Psi, \diamond\rangle$ being an undercut for $\langle\Gamma, \alpha\rangle$ means that $\{\gamma_i\} \cup (\Psi \setminus \{\gamma_i\}) \vdash \neg(\gamma_1 \wedge \ldots \wedge \gamma_n)$. By classical logic, $\Psi \setminus \{\gamma_i\} \vdash \neg(\gamma_1 \wedge \ldots \wedge \gamma_n)$ ensues and Ψ would not be a minimal subset of Δ entailing $\neg(\gamma_1 \wedge \ldots \wedge \gamma_n)$, a contradiction arises. Therefore, $\Psi \cap \Gamma = \emptyset$.
Reasoning by reductio ad absurdum again, assume $\Psi \subseteq X$. As $\langle\Psi, \diamond\rangle$ is an undercut for $\langle\Gamma, \alpha\rangle$, it follows that $\Psi \cup \Phi$ is inconsistent. $X \in N$ means that X is minimally inconsistent. Hence, $\Gamma \subseteq X$ and $\Psi \subseteq X$ imply $X = \Gamma \cup \Psi$. It has been shown above that Ψ and Γ are disjoint, therefore $\Psi = X \setminus \Gamma$ and a contradiction arises. Hence, the assumption must be false. So, $\Psi \not\subseteq X$. That is, there exists $\psi \in \Psi$ such that $\psi \notin X$.
Take Y to be $\Psi \cup \Gamma'$. (1) As $\Psi \cap \Gamma = \emptyset$ has been shown, $\Psi = Y \setminus \Gamma$. (2) $\Psi \cup \Gamma'$ has been shown to be minimally inconsistent, therefore $Y \in N$. (3) It has been shown that Γ' is non-empty, hence $\Gamma' \subseteq \Gamma$ yields $\Gamma \cap Y \neq \emptyset$. (4) Lastly, $\Gamma' \subseteq X \cap Y$ ensues from $\Gamma' \subseteq \Gamma \subseteq X$ and it follows that $X \cap Y$ is not empty because Γ' is not. Now, $\Psi \subseteq Y$ but it has been shown that $\Psi \not\subseteq X$ hence $X \neq Y$. An easy consequence of $X \cap Y \neq \emptyset$ and $X \neq Y$ is $(X, Y) \in A$. $\Box$

For an argument $\langle\Phi, \alpha\rangle$, the `Undercuts` algorithm is correct in the sense that it constructs a complete argument tree with $\langle\Phi, \alpha\rangle$ as the root. This is captured in Theorem 3, and it is based on the completeness result in Theorem 2, and the soundness results in Proposition 2 and Proposition 5.

Theorem 3. *For a knowledgebase Δ, a hypergraph* $\mathsf{Compilation}(\Delta) = (N, A)$*, and an argument* $\langle\Phi, \alpha\rangle$*, such that* $\Phi \subseteq \Delta$*, the algorithm* `Undercuts`$(\langle\Phi, \alpha\rangle, N, A)$ *returns the canonical undercuts for a complete argument tree* T *where the root of* T *is* $\langle\Phi, \alpha\rangle$*.*

Proof: From Proposition 2, we see that the call to the `FirstLevel` algorithm ensures that all the canonical undercuts for $\langle\Phi, \alpha\rangle$ are returned. From Proposition 2, each of the undercuts obtained via the `FirstLevel` algorithm are indeed canonical undercuts for $\langle\Phi, \alpha\rangle$.
From Proposition 2, we see that each call to the `Subcuts` algorithm returns all canonical undercuts to canonical undercuts, since given an undercut $\langle\Gamma, \diamond\rangle$ where $\Gamma \subset X$ for some $X \in N$, all undercuts $\langle Y \setminus \Gamma, \diamond\rangle$ are returned where $(X, Y) \in A$ and $Y \not\subseteq S$. The `Subcuts` algorithm is called by recursion to obtain all canonical undercuts to each canonical undercut by recursion. Furthermore, from Proposition 5, each of the undercuts obtained via the `Subcuts` algorithm are indeed canonical undercuts in T. $\Box$

We can consider the `Undercuts` algorithm, with the subsidiary algorithms `FirstLevel` and `Subcuts`, as constructing a depth-first search tree, annotating each node (except the root) of the tree with an undercut, and annotating each arc of the search tree with an arc from the hypergraph. In this way, the depth-first search tree is isomorphic with an argument tree rooted at $\langle\Phi, \alpha\rangle$.

Whilst the `Undercuts` algorithm, with the subsidiary algorithms `FirstLevel` and `Subcuts`, construct a depth-first search tree from the compilation, they do not backtrack. Furthermore, in order to construct an argument tree these algorithms only use constant time operations and tests, such as set membership, set union, and set intersection. In contrast, an algorithm for constructing an argument tree directly from an knowledgebase Δ has to search $\wp(\Delta)$ to find undercuts, and for each element $\Gamma \in \wp(\Delta)$, if an undercut with claim ψ is required, then the algorithm has to undertake the expensive tests of entailment (i.e. does $\Gamma \vdash \psi$ hold?), consistency (i.e. does $\Gamma \vdash \perp$ hold?), and minimality (i.e. does $\Gamma \cup \{\gamma_i\} \vdash \psi$ hold for each $\gamma_i \in \Gamma$?).

Constructing a compilation

A compilation of a knowledgebase can be constructed by the `GenerateCompilation` algorithm which in turn uses the `GenerateMinIncons` algorithm. These are defined below using the following subsidiary functions.

- `Subsets`(Δ, C) which returns the set of subsets of Δ that have a cardinality C;
- `NotSuperSet`(Φ, Θ) which is Boolean function that is true if there is no member of Θ that is a subset of Φ;
- `Inconsistent`(Φ) which is Boolean function that is true if $\Phi \vdash \perp$.

The `GenerateCompilation` algorithm takes the set of minimal inconsistent subsets of Δ, and then seeks all pairs of minimal inconsistent subsets with non-empty intersections, and hence giving the set of arcs of the graph.

Definition 10. *Given a set of formulae Δ, the* `GenerateCompilation` *algorithm finds the compilation for Δ.*

```
GenerateCompilation($\Delta$)
    $N$ = GenerateMinIncons($\Delta$)
    $M = N$
    $A = \emptyset$
    while $M \neq \emptyset$
        remove an arbitrary element $X$ from $M$
        let Temp $= M$
        while Temp $\neq \emptyset$
            remove an arbitrary element $Y$ from Temp
            if $X \cap Y \neq \emptyset$
            then $A = A \cup \{(X, Y)\}$
    return $(N, A)$
```

The `GenerateMinIncons` algorithm works by considering each subset of Δ as a possible minimal inconsistent subset of Δ. The search is started with the subsets of smallest cardinality, and then with each cycle of the outer while loop, the cardinality of the subsets is increased by one. The inner while loop checks each of the subsets Φ of a given cardinality. The if statement checks whether Φ is not a superset of a minimal inconsistent subset that has already been identified, and that Φ is inconsistent. If these conditions are met, then Φ is added to the set of minimal inconsistent subsets Θ that is eventually returned by the algorithm.

Definition 11. *Given a set of formulae Δ, the* `GenerateMinIncons` *algorithm returns all the minimal inconsistent subsets of Δ.*

```
GenerateMinIncons($\Delta$)
    $\Theta = \emptyset$
    $C = 1$
    while $C \leq |\Delta|$
        $\Omega$ = Subsets($\Delta, C$)
        while $\Omega \neq \emptyset$
            let $\Phi$ be an arbitrary member of $\Omega$
            if NotSuperSet($\Phi, \Theta$) & Inconsistent($\Phi$)
            then $\Theta = \Theta \cup \{\Phi\}$
            $\Omega = \Omega \setminus \{\Phi\}$
        $C = C + 1$
    return $\Theta$
```

We now turn to the complexity of constructing a compilation. We start by considering the potential size of $\mathsf{MinIncon}(\Delta)$. Let $\mathsf{Combinations}(x, y)$ be the number of combinations of subsets of cardinality y that can be formed from a set of cardinality x. In other words, $\mathsf{Combinations}(x, y) = x!/(x - y)!y!$. Also, let $\lfloor z \rfloor$ be the greatest integer less then or equal to z.

Proposition 7. *Let $n = |\Delta|$ and $m = \lfloor n/2 \rfloor$. If $\Delta \vdash \perp$, then* $1 \leq |\mathsf{MinIncon}(\Delta)| \leq \mathsf{Combinations}(n, m)$*.*

Proof For any $\Phi \in \mathsf{MinIncon}(\Delta)$, there is no $\Psi \in \mathsf{MinIncon}(\Delta)$ such that $\Phi \subset \Psi$ or $\Psi \subset \Phi$. The largest subset of $\wp(\Delta)$ for which no element is a subset of any other is the set of subsets of Δ that are of cardinality $\lfloor n/2 \rfloor$. The number of these subsets is given by $\mathsf{Combinations}(n, m)$. $\Box$

Whilst the above result shows that the cardinality of $\mathsf{MinIncon}(\Delta)$ may be an issue, in practice it is likely this cardinality will be significantly lower than the upper limit and so be of an acceptable and manageable size.

Proposition 8. *If* $|\Delta| = n$, *and* $m = \lfloor n/2 \rfloor$, *and* $k = \mathsf{Combinations}(n, m)$, *and* $\mathsf{Compilation}(\Delta) = (N, A)$, *then* $|N| \leq k$ *and* $|A| \leq (k \times (k-1))$.

From this, we see that the expensive part of compilation is the call $\texttt{GenerateMinIncons}(\Delta)$, which in the worst case involves 2^n tests where $|\Delta| = n$, and each test is of the following expensive form:

$$\texttt{NotSuperSet}(\Phi, \Theta)\ \&\ \texttt{Inconsistent}(\Phi)$$

In contrast, the algorithms for using the compilation are relatively inexpensive, involving only constant cost operations and tests such as set membership, set intersection, and set union. Hence, constructing a compilation is expensive, whereas the cost of using it is relatively inexpensive.

Discussion

Much progress has been made on developing formalisms for argumentation. Some algorithms for argumentation have been developed (for example (Kakas & Toni 1999; Cayrol, Doutre, & Mengin 2001; Baroni & Giacomin 2002; García & Simari 2004)). However, relatively little progress has been made in developing techniques for overcoming the computational challenges of constructing arguments.

In this paper, we have presented a compilation of a knowledgebase that uses the minimal inconsistent subsets of the knowledgebase, together with an algorithm for constructing undercuts for an argument, and by recursion, undercuts to undercuts. Compilation is expensive, but constructing argument trees for an given root argument is relatively inexpensive.

In future research, we will address both theoretical and empirical issues. At the theoretical level, we need to undertake a comprehensive computational complexity analysis of the underlying decision problems related to compilation and of the algorithms we have proposed here. These theoretical analyses may include positioning our approach to compilation within a general framework for complexity analyses for compilation techniques (Cadoli *et al.* 2002). At the empirical level, we need to implement the algorithms presented here, and undertake trials with sufficently large knowledgebases to investigate the viability and utility of this form of compilation.

Whilst our presentation is based on a particular approach to logic-based argumentation, we believe the proposal could be adapted for a range of other logic-based approaches to argumentation (for example (García & Simari 2004; Amgoud & Cayrol 2002)) by adapting our definition for minimal inconsistent subsets and our algorithm for using the compilation.

References

Amgoud, L., and Cayrol, C. 2002. A model of reasoning based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence* 34:197–216.

Baroni, P., and Giacomin, M. 2002. Argumentation through a distributed self-stabilizing approach. *Journal of Experimental and Theoretical Artificial Intelligence* 14(4):273–301.

Benferhat, S.; Dubois, D.; and Prade, H. 1993. Argumentative inference in uncertain and inconsistent knowledge bases. In *Proceedings of Uncertainty in Artificial Intelligence*, 1449–1445. Morgan Kaufmann.

Besnard, P., and Hunter, A. 2001. A logic-based theory of deductive arguments. *Artificial Intelligence* 128:203–235.

Besnard, P., and Hunter, A. 2005. Practical first-order argumentation. In *Proc. of the 20th National Conference on Artificial Intelligence (AAAI'2005)*, 590–595. MIT Press.

Boolos, G.; Burgess, J.; and Jeffrey, R. 2002. *Computability and Logic*. Cambridge University Press.

Cadoli, M.; Donini, F.; Liberatore, P.; and Schaerf, M. 2002. Preprocessing of intractable problems. *Information and Computation* 176(2):89–120.

Cayrol, C.; Doutre, S.; and Mengin, J. 2001. Dialectical proof theories for the credulous preferred semantics of argumentation frameworks. In *Quantitative and Qualitative Approaches to Reasoning with Uncertainty*, volume 2143 of *LNCS*, 668–679. Springer.

Chesnevar, C.; Maguitman, A.; and Loui, R. 2001. Logical models of argument. *ACM Computing Surveys* 32:337–383.

Chesnevar, C.; Simari, G.; and Godo, L. 2005. Computing dialectical trees efficiently in possibilistic defeasible logic programming. In *Proceedings of the 8th Interntational Logic Programming and Non-monotonic Reasoning Conference*, volume 3662 of *LNCS*, 158–171. Springer.

Dimopoulos, Y.; Nebel, B.; and Toni, F. 2002. On the computational complexity of assumption-based argumentation for default reasoning. *Artificial Intelligence* 141:57–78.

Dung, P.; Kowalski, R.; and Toni, F. 2006. Dialectic proof procedures for assumption-based, admissible argumentation. *Artificial Intelligence* 170(2):114–159.

Eiter, T., and Gottlob, G. 1995. The complexity of logic-based abduction. *Journal of the ACM* 42:3–42.

Elvang-Gøransson, M.; Krause, P.; and Fox, J. 1993. Dialectic reasoning with classically inconsistent information. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, 114–121. Morgan Kaufmann.

García, A., and Simari, G. 2004. Defeasible logic programming: An argumentative approach. *Theory and Practice of Logic Programming* 4(1):95–138.

Garey, M., and Johnson, D. 1979. *Computers and Intractability*. W H Freeman.

Haenni, R. 2001. Cost-bounded argumentation. *International Journal of Approximate Reasoning* 26:101–127.

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

Koriche, F. 2001. On anytime coherence-based reasoning. In *Quantitative and Qualitative Approaches to Rea-*

soning with Uncertainty, volume 2143 of *LNCS*, 556–567. Springer.

Koriche, F. 2002. Approximate coherence-based reasoning. *Journal of Applied Non-classical Logics* 12(2):239–258.

Levesque, H. 1984. A logic of implicit and explicit belief. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'84)*, 198–202.

Prakken, H., and Sartor, G. 1997. Argument-based extended logic programming with defeasible priorities. *Journal of Applied Non-Classical Logics* 7:25–75.

Prakken, H., and Vreeswijk, G. 2000. Logical systems for defeasible argumentation. In Gabbay, D., ed., *Handbook of Philosophical Logic*. Kluwer.

Schaerf, M., and Cadoli, M. 1995. Tractable reasoning via approximation. *Artificial Intelligence* 74:249–310.