Argument and Computation, 2014
Vol. 5, No. 1, 5–30, http://dx.doi.org/10.1080/19462166.2013.869765

Taylor & Francis
Taylor & Francis Group

Constructing argument graphs with deductive arguments: a tutorial

Philippe Besnard[a] and Anthony Hunter[b]*

[a]*IRIT, Université Paul Sabatier, F-31062 Toulouse, France;* [b]*Department of Computer Science, University College London, London WC1E 6BT, UK*

(*Received 1 November 2013; final version received 19 November 2013*)

A deductive argument is a pair where the first item is a set of premises, the second item is a claim, and the premises entail the claim. This can be formalised by assuming a logical language for the premises and the claim, and logical entailment (or consequence relation) for showing that the claim follows from the premises. Examples of logics that can be used include classical logic, modal logic, description logic, temporal logic, and conditional logic. A counterargument for an argument A is an argument B where the claim of B contradicts the premises of A. Different choices of logic, and different choices for the precise definitions of argument and counterargument, give us a range of possibilities for formalising deductive argumentation. Further options are available to us for choosing the arguments and counterarguments we put into an argument graph. If we are to construct an argument graph based on the arguments that can be constructed from a knowledgebase, then we can be exhaustive in including all arguments and counterarguments that can be constructed from the knowledgebase. But there are other options available to us. We consider some of the possibilities in this review.

Keywords: structured argumentation; logic-based argumentation; deductive arguments

1. Introduction

Abstract argumentation, as proposed by Dung (1995), provides a good starting point for formalising argumentation. Dung proposed that a set of arguments and counterarguments could be represented by a directed graph. Each node in the graph denotes an argument and each arc denotes one argument attacking another. So if there is an arc from node A to node B, then A attacks B, or equivalently A is a counterargument to B. See Figure 1 for an example of an abstract argument graph.

Even though abstract argumentation provides a clear and precise approach to formalising aspects of argumentation, the arguments are treated as atomic. There is no formalised content to an argument, and so all arguments are treated as equal. Therefore, if we want to understand individual arguments, we need to provide content for them. This leads to the idea of "instantiating" abstract argumentation with deductive arguments. Each deductive argument has some premises from which a claim is derived by deductive reasoning.

In deductive reasoning, we start with some premises, and we derive a conclusion using one or more inference steps. Each inference step is infallible in the sense that it does not introduce uncertainty. In other words, if we accept the premises are valid, then we should accept that the intermediate conclusion of each inference step is valid, and therefore we should accept that the conclusion is valid. For example, if we accept that Philippe and Tony are having tea together in London is valid, then we should accept that Philippe is not in Toulouse (assuming the background knowledge that London and Toulouse are different places, and that nobody can be in different places at the same time). As another example, if we accept that Philippe and Tony are having an

*Corresponding author. Email: anthony.hunter@ucl.ac.uk, a.hunter@cs.ucl.ac.uk

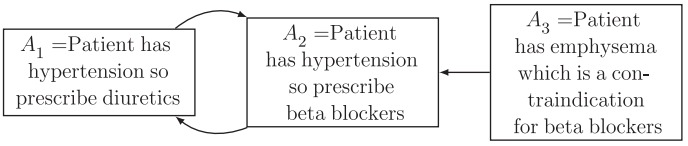


Figure 1. Example of an abstract argument graph which captures a decision-making scenario where there are two alternatives for treating a patient, diuretics or beta blockers. Since only one treatment should be given for the disorder, each argument attacks the other. There is also a reason to not give beta blockers, as the patient has emphysema which is a contraindication for this treatment.

ice cream together in Toulouse is valid, then we should accept that Tony is not in London. Note, however, we do not need to believe or know that the premises are valid to apply deductive reasoning. Rather, deductive reasoning allows us to obtain conclusions that we can accept contingent on the validity of their premises. So for the first example above, the reader might not know whether or not Philippe and Tony are having tea together in London. However, the reader can accept that Philippe is not in Toulouse, contingent on the validity of these premises. Important alternatives to deductive reasoning in argumentation include inductive reasoning, abductive reasoning, and analogical reasoning.

In this tutorial, we assume that deductive reasoning is formalised by a monotonic logic. Each deductive argument is a pair where the first item is a set of premises that logically entails the second item according to the choice of monotonic logic. So we have a logical language to express the set of premises, and the claim, and we have a logical consequence relation to relate the premises to the claim.

Key benefits of deductive arguments include: (1) explicit representation of the information used to support the claim of the argument; (2) explicit representation of the claim of the argument; and (3) a simple and precise connection between the support and claim of the argument via the consequence relation. What a deductive argument does not provide is a specific proof of the claim from the premises. There may be more than one way of proving the claim from the premises, but the argument does not specify which is used. It is, therefore, indifferent to the proof used.

Deductive argumentation is formalised in terms of deductive arguments and counterarguments, and there are various choices for defining this (Besnard & Hunter 2008). Deductive argumentation offers a simple route to instantiating abstract argumentation which we will consider in this tutorial paper. Perhaps the first paper to consider this is by Cayrol who instantiated Dung's proposal with deductive arguments based on classical logic (Cayrol 1995).

In the rest of this tutorial, we will investigate some of the choices we have for defining arguments and counterarguments, and for how they can be used in modelling argumentation. We will focus on two choices for base logic. These are simple logic (which has a language of literals and rules of the form $\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta$, where $\alpha_1, \ldots, \alpha_n, \beta$ are literals, and modus ponens is the only proof rule) and classical logic (propositional and first-order classical logic). Then for instantiating argument graphs (i.e. for specifying what the arguments and attacks are in an argument graph), we will consider descriptive graphs and generative graphs defined informally as follows.

- *Descriptive graphs.* Here we assume that the structure of the argument graph is given, and the task is to identify the premises and claim of each argument. Therefore, the input is an abstract argument graph, and the output is an instantiated argument graph. This kind of task arises in many situations. For example, if we are listening to a debate, we hear the arguments exchanged and can construct the instantiated argument graph to reflect the debate.
- *Generative graphs.* Here we assume that we start with a knowledgebase (i.e. a set of logical formula), and the task is to generate the arguments and counterarguments (and hence the

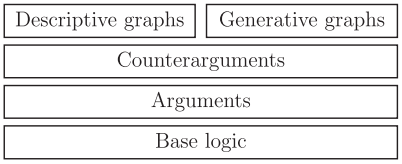


Figure 2. Framework for constructing argument graphs with deductive arguments: for defining a specific argumentation system, there are four levels for the specification: (1) a base logic is required for defining the logical language and the consequence or entailment relation (i.e. what inferences follow from a set of formulae); (2) a definition of an argument $\langle \Phi, \alpha \rangle$ specified using the base logic (e.g. Φ is consistent and Φ entails α); (3) a definition of counterargument specified using the base logic (i.e. a definition for when one argument attacks another); and (4) a definition of how the arguments and counterarguments are composed into an argument graph (which is either a descriptive graph or some form of generative graph).

attacks between arguments). Therefore, the input is a knowledgebase, and the output is an instantiated argument graph. This kind of task also arises in many situations: for example, if we are making a decision based on conflicting information. We have various items of information that we represent by formulae in the knowledgebase, and we construct an instantiated argument graph to reflect the arguments and counterarguments that follow from that information.

For constructing both descriptive graphs and generative graphs, there may be a dynamic aspect to the process. For instance, when constructing descriptive graphs, we may be unsure of the exact structure of the argument graph, and it is only by instantiating individual arguments that we are able to say whether it is attacked or attacks another argument. As another example, when constructing generative graphs, we may be involved in a dialogue, and so through the dialogue, we may obtain further information which allows us to generate further arguments that can be added to the argument graph.

So in order to construct argument graphs with deductive arguments, we need to specify the choice of logic (which we call the base logic) that we use to define arguments and counterarguments, the definition for arguments, the definition for counterarguments, and the definition for instantiating argument graphs. For the latter, we can either produce a descriptive graph or a generative graph. We will explore various options for generative graphs. We summarise the framework for constructing argument graphs with deductive arguments in Figure 2.

We proceed as follows: in Section 2, we briefly review the definitions for abstract argumentation; in Section 3, we consider options for arguments in deductive argumentation; in Section 4, we consider options for counterarguments in deductive argumentation; in Section 5, we consider options for constructing argument graphs instantiated with deductive arguments; in Section 6, we briefly compare the approach expounded in this tutorial to other approaches to structured argumentation; and in Section 7, we discuss the approach of deductive argumentation and provide suggestions for further reading.

2. Abstract argumentation

An *abstract argument graph* is a pair $(\mathcal{A}, \mathcal{R})$, where $\mathcal{A}$ is a set and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$. Each element $A \in \mathcal{A}$ is called an *argument* and $(A, B) \in \mathcal{R}$ means that A *attacks* B (accordingly, A is said to be an *attacker* of B) and so A is a *counterargument* for B. A set of arguments $S \subseteq \mathcal{A}$ *attacks* $A_j \in \mathcal{A}$ iff there is an argument $A_i \in S$ such that A_i attacks A_j. Also, S *defends* $A_i \in \mathcal{A}$ iff for each argument $A_j \in \mathcal{A}$, if A_j attacks A_i then S attacks A_j. A set $S \subseteq \mathcal{A}$ of arguments is *conflict-free* iff there are

no arguments A_i and A_j in S such that A_i attacks A_j. Let Γ be a conflict-free set of arguments, and let $\mathsf{Defended} : \wp(\mathcal{A}) \rightarrow \wp(\mathcal{A})$ be a function such that $\mathsf{Defended}(\Gamma) = \{A \mid \Gamma \text{ defends } A\}$. We consider the following extensions: (1) Γ is a *complete extension* iff $\Gamma = \mathsf{Defended}(\Gamma)$; (2) Γ is a *grounded extension* iff it is the minimal (w.r.t. set inclusion) complete extension; (3) Γ is a *preferred extension* iff it is a maximal (w.r.t. set inclusion) complete extension; and (4) Γ is a *stable extension* iff it is a preferred extension that attacks every argument that is not in the extension.

Some argument graphs can be large, and yet we might only be interested in whether some subset of the arguments is in an extension according to some semantics. For this, we introduce the following definitions that lead to the notion of a focal graph.

DEFINITION 2.1 Let $G = (\mathcal{A}, \mathcal{R})$ be an argument graph. An argument graph $(\mathcal{A}', \mathcal{R}')$ is *faithful* with respect to $(\mathcal{A}, \mathcal{R})$ iff $(\mathcal{A}', \mathcal{R}')$ is a subgraph of $(\mathcal{A}, \mathcal{R})$ and for all arguments $A_i, A_j \in \mathcal{A}$, if $A_j \in \mathcal{A}'$ and $(A_i, A_j) \in \mathcal{R}$, then $A_i \in \mathcal{A}'$, and $\mathcal{R}' = \{(A_i, A_j) \mid \mathcal{R} \mid A_i, A_j \in \mathcal{A}'\}$.

EXAMPLE 2.2 Consider the following graph G

There are three subgraphs that are faithful with respect to G: (1) the graph G ; (2) the subgraph containing just the argument A_1; and (3) the following subgraph. All other subgraphs of G are not faithful.

A faithful subgraph has the same extensions as the graph modulo the arguments in the subgraph. So for every argument A in the subgraph, if A is in the grounded extension in the subgraph, then A is in the grounded extension of the graph, and vice versa. Similarly, for every argument A in the subgraph, if A is in a preferred extension of the subgraph, then A is in a preferred extension of the graph, and vice versa. This follows directly from the directionality criterion of Baroni & Giacomin (2007) that says that for a subgraph, arguments in the graph that do not attack any arguments in the subgraph have no effect on the extensions of the subgraph. Therefore, we can ignore the arguments that are not in a faithful subgraph.

DEFINITION 2.3 Let $\Pi \subseteq \mathcal{A}$ be a set of arguments of interest called the *focus*. A graph $(\mathcal{A}', \mathcal{R}')$ is the *focal graph* of graph $(\mathcal{A}, \mathcal{R})$ with respect to focus Π iff $(\mathcal{A}', \mathcal{R}')$ is the smallest subgraph of $(\mathcal{A}, \mathcal{R})$ such that $\Pi \subseteq \mathcal{A}'$ and $(\mathcal{A}', \mathcal{R}')$ is faithful with respect to $(\mathcal{A}, \mathcal{R})$.

EXAMPLE 2.4 Continuing Example 2.2, if we let $\Pi = \{A_1, A_2\}$ be the focus, then the third subgraph (i.e. the faithful graph containing A_1, A_2, and A_3) is the focal graph.

The motivation for finding the focal graph is that given a set of arguments Π as the focus, we want to just have those arguments and any arguments that may affect whether or not any of the arguments in Π are in an extension. By taking the directionality of the arcs into account (i.e. the directionality criteria; Baroni & Giacomin 2007; Liao, Jin, & Koons 2011), we can ignore the other arguments.

3. Arguments

Each argument in deductive argumentation is defined using a logic which we call the *base logic*. In this paper, we focus on two options for the base logic, namely simple logic and classical logic, but other options include conditional logic, temporal logic, and paraconsistent logic.

Let $\mathcal{L}$ be a language for a logic, and let $\vdash_i$ be the consequence relation for that logic. Therefore, $\vdash_i \subseteq \wp(\mathcal{L}) \times \mathcal{L}$. If α is an atom in $\mathcal{L}$, then α is a *positive literal* in $\mathcal{L}$ and $\neg\alpha$ is a *negative literal* in $\mathcal{L}$. For a literal β, the *complement* of β is defined as follows: if β is a positive literal, i.e. it is of the form α, then the complement of β is the negative literal $\neg\alpha$, and if β is a negative literal, i.e. it is of the form $\neg\alpha$, then the complement of β is the positive literal α.

A *deductive argument* is an ordered pair $\langle \Phi, \alpha \rangle$, where $\Phi \vdash_i \alpha$. Φ is the support, or premises, or assumptions of the argument, and α is the claim, or conclusion, of the argument. The definition for a deductive argument only assumes that the premises entail the claim (i.e. $\Phi \vdash_i \alpha$). For an argument $A = \langle \Phi, \alpha \rangle$, the function $\mathsf{Support}(A)$ returns Φ and the function $\mathsf{Claim}(A)$ returns α.

Many proposals have further constraints for an ordered pair $\langle \Phi, \alpha \rangle$ to be an argument. The most commonly assumed constraint is the *consistency constraint*: an argument $\langle \Phi, \alpha \rangle$ satisfies this constraint when Φ is consistent. For richer logics, such as classical logic, consistency is often regarded as a desirable property of a deductive argument because claims that are obtained with logics such as classical logic from inconsistent premises are normally useless as illustrated in the next example.

EXAMPLE 3.1 If we assume the consistency constraint, then the following are not arguments.

$$\langle \{\texttt{study(Sid, logic)}, \neg\texttt{study(Sid, logic)}\},$$
$$\texttt{study(Sid, logic)} \leftrightarrow \neg\texttt{study(Sid, logic)}\rangle$$
$$\langle \{\texttt{study(Sid, logic)}, \neg\texttt{study(Sid, logic)}\}, \texttt{KingOfFrance(Sid)}\rangle$$

In contrast, for weaker logics (such as paraconsistent logics), it may be desirable to not impose the consistency constraint. With such logics, a credulous approach could be taken so that pros and cons could be obtained from inconsistent premises (as illustrated by the following example).

EXAMPLE 3.2 If we assume the base logic is a paraconsistent logic (such as Belnap's four-valued logic), and we do not impose the consistent constraint, then the following are arguments.

$$\langle \{\texttt{study(Sid, logic)} \wedge \neg\texttt{study(Sid, logic)}\}, \texttt{study(Sid, logic)}\rangle$$
$$\langle \{\texttt{study(Sid, logic)} \wedge \neg\texttt{study(Sid, logic)}\}, \neg\texttt{study(Sid, logic)}\rangle$$

Another commonly assumed constraint is the *minimality constraint*: an argument $\langle \Phi, \alpha \rangle$ satisfies this constraint when there is no $\Psi \subset \Phi$ such that $\Psi \vdash \alpha$. Minimality is often regarded as a desirable property of a deductive argument because it eliminates irrelevant premises (as in the following example).

EXAMPLE 3.3 If we assume the minimality constraint, then the following is not an argument.

$$\langle \{\texttt{report(rain)}, \texttt{report(rain)} \rightarrow \texttt{carry(umbrella)}, \texttt{happy(Sid)}\},$$
$$\texttt{carry(umbrella)}\rangle$$

When we construct a knowledgebase, with simple logic, classical logic, or other base logics, it is possible that some or all of the formulae could be incorrect. For instance, individual formulae may

come from different and conflicting sources, they may reflect options that disagree and represent uncertain information. A knowledgebase may be inconsistent, and individual formulae may be contradictory. After all, if the knowledge is not inconsistent (i.e. it is consistent), then we will not have counterarguments. We may also include formulae that we know are not always correct. For instance, we may include a formula such as the following that says that a water sample taken from the Mediterranean sea in summer will be above 15°C. Whilst this may be a useful general rule, it is not always true. For instance, the sample could be taken when there is a period of bad weather, or the sample is taken from a depth of over 500 m.

$$\forall \mathtt{X}, \mathtt{Y}.\mathtt{watersample}(\mathtt{X}) \wedge \mathtt{location}(\mathtt{X}, \mathtt{Mediterranean}) \wedge \mathtt{season}(\mathtt{X}, \mathtt{summer}) \wedge \mathtt{temperature}(\mathtt{X}, \mathtt{Y}) \rightarrow \mathtt{Y} > 15$$

In the following subsections, we define arguments based on simple logic and on classical logic as the base logic. Alternative base logics include description logic, paraconsistent logic, temporal logic, and conditional logic.

3.1. *Arguments based on simple logic*

Simple logic is based on a language of literals and simple rules where each *simple rule* is of the form $\alpha_1 \wedge \cdots \wedge \alpha_k \rightarrow \beta$, where α_1 to α_k and β are literals. A *simple logic knowledgebase* is a set of literals and a set of simple rules. The consequence relation is modus ponens (i.e. implication elimination) as defined next.

DEFINITION 3.4 The *simple consequence relation*, denoted $\vdash_s$, which is the smallest relation satisfying the following condition, and where Δ is a simple logic knowledgebase: $\Delta \vdash_s \beta$ iff there is an $\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta \in \Delta$, and for each $\alpha_i \in \{\alpha_1, \ldots, \alpha_n\}$, either $\alpha_i \in \Delta$ or $\Delta \vdash_s \alpha_i$.

EXAMPLE 3.5 Let $\Delta = \{a, b, a \wedge b \rightarrow c, c \rightarrow d\}$. Hence, $\Delta \vdash_s c$ and $\Delta \vdash_s d$. However, $\Delta \nvdash_s a$ and $\Delta \nvdash_s b$.

DEFINITION 3.6 Let Δ be a simple logic knowledgebase. For $\Phi \subseteq \Delta$, and a literal α, $\langle \Phi, \alpha \rangle$ is a *simple argument* iff $\Phi \vdash_s \alpha$ and there is no proper subset Φ' of Φ such that $\Phi' \vdash_s \alpha$.

So each simple argument is minimal, but not necessarily consistent (where consistency for a simple logic knowledgebase Δ means that for no atom α does $\Delta \vdash_s \alpha$ and $\Delta \vdash_s \neg\alpha$ hold). We do not impose the consistency constraint in the definition for simple arguments as simple logic is paraconsistent, and therefore can support a credulous view on the arguments that can be generated.

EXAMPLE 3.7 Let $\mathtt{p}_1$, $\mathtt{p}_2$, and $\mathtt{p}_3$ be the following formulae. Then $\langle \{\mathtt{p}_1, \mathtt{p}_2, \mathtt{p}_3\}, \mathtt{goodInvestment}(\mathtt{BP}) \rangle$ is a simple argument. Note, we use $\mathtt{p}_1$, $\mathtt{p}_2$, and $\mathtt{p}_3$ as labels in order to make the presentation of the premises more concise.

$$\mathtt{p}_1 = \mathtt{oilCompany}(\mathtt{BP})$$

$$\mathtt{p}_2 = \mathtt{goodPerformer}(\mathtt{BP})$$

$$\mathtt{p}_3 = \mathtt{oilCompany}(\mathtt{BP}) \wedge \mathtt{goodPerformer}(\mathtt{BP})) \rightarrow \mathtt{goodInvestment}(\mathtt{BP})$$

Simple logic is a practical choice as a base logic for argumentation. Having a logic with simple rules and modus ponens is useful for applications because the behaviour is quite predictable

in the sense that given a knowledgebase it is relatively easy to anticipate the inferences that come from the knowledgebase. Furthermore, it is relatively easy to implement an algorithm for generating the arguments and counterarguments from a knowledgebase. The downside of simple logic as a base logic is that the proof theory is weak. It only incorporates modus ponens (i.e. implication elimination) and so many useful kinds of reasoning (e.g. contrapositive reasoning) are not supported.

3.2. *Arguments based on classical logic*

Classical logic is appealing as the choice of base logic as it better reflects the richer deductive reasoning often seen in arguments arising in discussions and debates.

We assume the usual propositional and predicate (first-order) languages for classical logic, and the usual the *classical consequence relation*, denoted $\vdash$. A *classical knowledgebase* is a set of classical propositional or predicate formulae.

DEFINITION 3.8 For a classical knowledgebase Φ, and a classical formula α, $\langle \Phi, \alpha \rangle$ is a *classical argument* iff $\Phi \vdash \alpha$ and $\Phi \nvdash \bot$ and there is no proper subset Φ' of Φ such that $\Phi' \vdash \alpha$.

So a classical argument satisfies both minimality and consistency. We impose the consistency constraint because we want to avoid the useless inferences that come with inconsistency in classical logic (such as via ex falso quodlibet).

EXAMPLE 3.9 The following classical argument uses a universally quantified formula in contrapositive reasoning to obtain the following claim about number 77.

$$\langle \{\forall \mathtt{X}.\mathtt{multipleOfTen}(\mathtt{X}) \rightarrow \mathtt{even}(\mathtt{X}), \neg \mathtt{even}(77)\}, \neg \mathtt{multipleOfTen}(77) \rangle$$

Given the central role classical logic has played in philosophy, linguistics, and computer science (software engineering, formal methods, data and knowledge engineering, artificial intelligence, computational linguistics, etc.), we should consider how it can be used in argumentation. Classical propositional logic and classical predicate logic are expressive formalisms which capture deeper insights about the world than is possible with restricted formalisms such as simple logic.

4. Counterarguments

A counterargument is an argument that attacks another argument. In deductive argumentation, we define the notion of counterargument in terms of logical contradiction between the claim of the counterargument and the premises of claim of the attacked argument. We explore some of the kinds of counterargument that can be specified for simple logic and classical logic.

4.1. *Counterarguments based on simple logic*

For simple logic, we consider two forms of counterargument. For this, recall that literal α is the complement of literal β if and only if α is an atom and β is $\neg\alpha$ or if β is an atom and α is $\neg\beta$.

DEFINITION 4.1 For simple arguments A and B, we consider the following type of *simple attack*:

- A is a *simple undercut* of B if there is a simple rule $\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta$ in $\mathsf{Support}(B)$ and there is an $\alpha_i \in \{\alpha_1, \ldots, \alpha_n\}$ such that $\mathsf{Claim}(A)$ is the complement of α_i.
- A is a *simple rebut* of B if $\mathsf{Claim}(A)$ is the complement of $\mathsf{Claim}(B)$.

EXAMPLE 4.2 The first argument A_1 captures the reasoning that the metro is an efficient form of transport, so one can use it. The second argument A_2 captures the reasoning that there is a strike on the metro, and so the metro is not an efficient form of transport (at least on the day of the strike). A_2 undercuts A_1.

$$A_1 = \langle \{\texttt{efficientMetro}, \texttt{efficientMetro} \rightarrow \texttt{useMetro}\}, \texttt{useMetro} \rangle$$

$$A_2 = \langle \{\texttt{strikeMetro}, \texttt{strikeMetro} \rightarrow \neg\texttt{efficientMetro}\}, \neg\texttt{efficientMetro} \rangle$$

EXAMPLE 4.3 The first argument A_1 captures the reasoning that the government has a budget deficit, and so the government should cut spending. The second argument A_2 captures the reasoning that the economy is weak, and so the government should not cut spending. The arguments rebut each other.

$$A_1 = \langle \{\texttt{govDeficit}, \texttt{govDeficit} \rightarrow \texttt{cutGovSpending}\}, \texttt{cutGovSpending} \rangle$$

$$A_2 = \langle \{\texttt{weakEconomy}, \texttt{weakEconomy} \rightarrow \neg\texttt{cutGovSpending}\}, \neg\texttt{cutGovSpending} \rangle$$

So in simple logic, a rebut attacks the claim of an argument, and an undercut attacks the premises of the argument (either by attacking one of the literals or by attacking the consequent of one of the rules in the premises).

Simple arguments and counterarguments can be used to model defeasible reasoning. For this, we use simple rules that are normally correct but sometimes incorrect. For instance, if Sid has the goal of going to work, Sid takes the metro. This is generally true, but sometimes Sid works at home, and so it is no longer true that Sid takes the metro, as we see in the next example.

EXAMPLE 4.4 The first argument A_1 captures the general rule that if `workDay` holds, then `useMetro(Sid)` holds. The use of the simple rule in A_1 requires that the assumption `normal` holds. This is given as an assumption. The second argument A_2 undercuts the first argument by contradicting the assumption that `normal` holds

$$A_1 = \langle \{\texttt{workDay}, \texttt{normal}, \texttt{workDay} \wedge \texttt{normal} \rightarrow \texttt{useMetro(Sid)}\}, \texttt{useMetro(Sid)} \rangle$$

$$A_2 = \langle \{\texttt{workAtHome(Sid)}, \texttt{workAtHome(Sid)} \rightarrow \neg\texttt{normal}\}, \neg\texttt{normal} \rangle$$

If we start with just argument A_1, then A_1 is undefeated, and so `useMetro(Sid)` is an acceptable claim. However, if we then add A_2, then A_1 is a defeated argument and A_2 is an undefeated argument. Hence, if we have A_2, we have to withdraw `useMetro(Sid)` as an acceptable claim.

So by having appropriate conditions in the antecedent of a simple rule we can disable the rule by generating a counterargument that attacks it. This in effect stops the usage of the simple rule. This means that we have a convention to attack an argument based on the inferences obtained by the simple logic (e.g. as in Examples 4.2 and 4.3), or on the rules used (e.g. Example 4.4).

This way to disable rules by adding appropriate conditions (as in Example 4.4) is analogous to the use abnormality predicates used in formalisms such as circumscription (see, for example, McCarthy 1980). We can use the same approach to capture defeasible reasoning in other logics such as classical logic. Note, this does not mean that we turn the base logic into a nonmonotonic logic. Both simple logic and classical logic are monotonic logics. Hence, for a simple logic knowledgebase Δ (and similarly for a classical logic knowledgebase Δ), the set of simple arguments (respectively, classical arguments) obtained from Δ is a subset of the set of simple arguments (respectively, classical arguments) obtained from $\Delta \cup \{\alpha\}$, where α is a formula not in Δ. But at the level of evaluating arguments and counterarguments, we have non-monotonic defeasible

behaviour. For instance in Example 4.2, with just A_1 we have the acceptable claim `useMetro`, but then when we have also A_2, we have to withdraw this claim. In other words, if the set of simple arguments is $\{A_1\}$, then we can construct an argument graph with just A_1, and by applying Dung's dialectical semantics, there is one extension containing A_1. However, if the set of simple arguments is $\{A_1, A_2\}$, then we can construct an argument graph with A_1 attacked by A_2, and by applying Dung's dialectical semantics, there is one extension containing A_2. This illustrates the fact that the argumentation process is non-monotonic.

4.2. *Counterarguments based on classical logic*

Given the expressivity of classical logic (in terms of language and inferences), there are a number of natural ways that we can define counterarguments.

DEFINITION 4.5 Let A and B be two classical arguments. We define the following types of *classical attack*.

- A is a *classical defeater* of B if $\mathsf{Claim}(A) \vdash \neg \bigwedge \mathsf{Support}(B)$.
- A is a *classical direct defeater* of B if $\exists \phi \in \mathsf{Support}(B)$ s.t. $\mathsf{Claim}(A) \vdash \neg\phi$.
- A is a *classical undercut* of B if $\exists \Psi \subseteq \mathsf{Support}(B)$ s.t. $\mathsf{Claim}(A) \equiv \neg \bigwedge \Psi$.
- A is a *classical direct undercut* of B if $\exists \phi \in \mathsf{Support}(B)$ s.t. $\mathsf{Claim}(A) \equiv \neg\phi$.
- A is a *classical canonical undercut* of B if $\mathsf{Claim}(A) \equiv \neg \bigwedge \mathsf{Support}(B)$.
- A is a *classical rebuttal* of B if $\mathsf{Claim}(A) \equiv \neg\mathsf{Claim}(B)$.
- A is a *classical defeating rebuttal* of B if $\mathsf{Claim}(A) \vdash \neg\mathsf{Claim}(B)$.

To illustrate these different notions of classical counterargument, we consider the following examples, and we relate these definitions in Figure 3 where we show that classical defeaters are the most general of these definitions.

EXAMPLE 4.6 Let $\Delta = \{a \vee b, a \leftrightarrow b, c \rightarrow a, \neg a \wedge \neg b, a, b, c, a \rightarrow b, \neg a, \neg b, \neg c\}$

$\langle\{a \vee b, c\}, (a \vee b) \wedge c\rangle$is a classical defeater of$\langle\{\neg a, \neg b\}, \neg a \wedge \neg b\rangle$

$\langle\{a \vee b, c\}, (a \vee b) \wedge c\rangle$is a classical direct defeater of$\langle\{\neg a \wedge \neg b\}, \neg a \wedge \neg b\rangle$

$\langle\{\neg a \wedge \neg b\}, \neg(a \wedge b)\rangle$is a classical undercut of$\langle\{a, b, c\}, a \wedge b \wedge c\rangle$

$\langle\{\neg a \wedge \neg b\}, \neg a\rangle$is a classical direct undercut of$\langle\{a, b, c\}, a \wedge b \wedge c\rangle$

$\langle\{\neg a \wedge \neg b\}, \neg(a \wedge b \wedge c)\rangle$is a classical canonical undercut of$\langle\{a, b, c\}, a \wedge b \wedge c\rangle$

$\langle\{a, a \rightarrow b\}, b \vee c\rangle$is a classical rebuttal of$\langle\{\neg a \wedge \neg b, \neg c\}, \neg(b \vee c)\rangle$

$\langle\{a, a \rightarrow b\}, b\rangle$is a classical defeating rebuttal of$\langle\{\neg a \wedge \neg b, \neg c\}, \neg(b \vee c)\rangle$

Using simple logic, the definitions for counterarguments against the support of another argument are limited to attacking just one of the items in the support. In contrast, using classical logic, a counterargument can be against more than one item in the support. For example, in Example 4.7, the undercut is not attacking an individual premise but rather saying that two of the premises are incompatible (in this case that the premises `lowCostFly` and `luxuryFly` are incompatible).

EXAMPLE 4.7 Consider the following arguments. A_1 is attacked by A_2 as A_2 is an undercut of A_1 though it is neither a direct undercut nor a canonical undercut. Essentially, the attack says that the

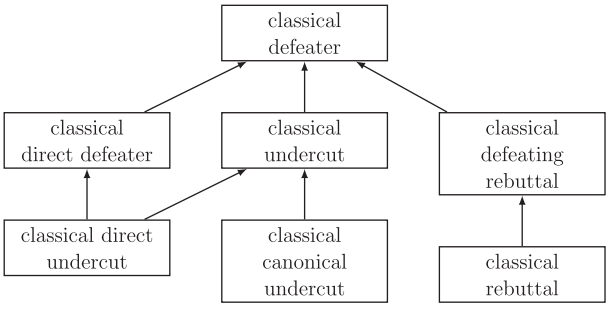


Figure 3. We can represent the containment between the classical attack relations as above where an arrow from R_1 to R_2 indicates that $R_1 \subseteq R_2$. For proofs, see Besnard & Hunter (2001) and Gorogiannis & Hunter (2011).

flight cannot be both a low cost flight and a luxury flight.

$A_1 = \langle \{\texttt{lowCostFly}, \texttt{luxuryFly}, \texttt{lowCostFly} \wedge \texttt{luxuryFly} \rightarrow \texttt{goodFly}\}, \texttt{goodFly} \rangle$

$A_2 = \langle \{\neg\texttt{lowCostFly} \vee \neg\texttt{luxuryFly}\}, \neg\texttt{lowCostFly} \vee \neg\texttt{luxuryFly} \rangle$

Trivially, undercuts are defeaters but it is also quite simple to establish that rebuttals are defeaters. Furthermore, if an argument has defeaters then it has undercuts. It may happen that an argument has defeaters but no rebuttals as illustrated next.

EXAMPLE 4.8 Let $\Delta = \{\neg\texttt{containsGarlic} \wedge \texttt{goodDish}, \neg\texttt{goodDish}\}$. Then the following argument has at least one defeater but no rebuttal.

$$\langle \{\neg\texttt{containsGarlic} \wedge \texttt{goodDish}\}, \neg\texttt{containsGarlic} \rangle$$

There are some important differences between rebuttals and undercuts that can be seen in the following examples.

EXAMPLE 4.9 Consider the following arguments. The first argument A_1 is a direct undercut to the second argument A_2, but neither rebuts each other. Furthermore, A_1 "agrees" with the claim of A_2 since the premises of A_1 could be used for an alternative argument with the same claim as A_2.

$A_1 = \langle \{\neg\texttt{containsGarlic} \wedge \neg\texttt{goodDish}\}, \neg\texttt{containsGarlic} \rangle$

$A_2 = \langle \{\texttt{containsGarlic}, \texttt{containtsGarlic} \rightarrow \neg\texttt{goodDish}\}, \neg\texttt{goodDish} \rangle$

EXAMPLE 4.10 Consider the following arguments. The first argument is a rebuttal of the second argument, but it is not an undercut because the claim of the first argument is not equivalent to the negation of some subset of the premises of the second argument.

$A_1 = \langle \{\texttt{goodDish}\}, \texttt{goodDish} \rangle$

$A_2 = \langle \{\texttt{containsGarlic}, \texttt{containtsGarlic} \rightarrow \neg\texttt{goodDish}\}, \neg\texttt{goodDish} \rangle$

So an undercut for an argument need not be a rebuttal for that argument, and a rebuttal for an argument need not be an undercut for that argument.

Arguments are not necessarily independent. In a sense, some encompass others (possibly up to some form of equivalence), which is the topic we now turn to.

DEFINITION 4.11 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 4.12 $\langle \{a\}, a \vee b \rangle$ is more conservative than $\langle \{a, a \rightarrow b\}, b \rangle$.

Roughly speaking, a more conservative argument is more general: it is, so to speak, less demanding on the support and less specific about the claim.

EXAMPLE 4.13 Consider the following formulae.

$$\mathtt{p}_1 = \mathtt{divisibleByTen(50)}$$
$$\mathtt{p}_2 = \forall \mathtt{X.divisibleByTen(X)} \rightarrow \mathtt{divisibleByTwo(X)}$$
$$\mathtt{p}_3 = \forall \mathtt{X.divisibleByTwo(X)} \rightarrow \mathtt{even(X)}$$

Hence, A_1 is an argument with the claim "The number 50 is divisible by 2", and A_2 is an argument with the claim "The number 50 is divisible by 2 and the number 50 is an even number."However, A_1 is more conservative than A_2.

$$A_1 = \langle \{\mathtt{p}_1, \mathtt{p}_2\}, \mathtt{divisibleByTwo(50)} \rangle$$
$$A_2 = \langle \{\mathtt{p}_1, \mathtt{p}_2, \mathtt{p}_3, \}, \mathtt{even(50)} \wedge \mathtt{divisibleByTwo(50)} \rangle$$

We can use the notion of "more conservative" to help us identify the most useful counterarguments amongst the potentially large number of counterarguments.

EXAMPLE 4.14 Let $\{a, b, c, \neg a \vee \neg b \vee \neg c\}$ be our knowledgebase. Suppose we start with the argument $\langle \{a, b, c\}, a \wedge b \wedge c \rangle$. Now we have numerous undercuts to this argument including the following.

$$\langle \{b, c, \neg a \vee \neg b \vee \neg c\}, \neg a \rangle$$
$$\langle \{a, c, \neg a \vee \neg b \vee \neg c\}, \neg b \rangle$$
$$\langle \{a, b, \neg a \vee \neg b \vee \neg c\}, \neg c \rangle$$
$$\langle \{c, \neg a \vee \neg b \vee \neg c\}, \neg a \vee \neg b \rangle$$
$$\langle \{b, \neg a \vee \neg b \vee \neg c\}, \neg a \vee \neg c \rangle$$
$$\langle \{a, \neg a \vee \neg b \vee \neg c\}, \neg b \vee \neg c \rangle$$
$$\langle \{\neg a \vee \neg b \vee \neg c\}, \neg a \vee \neg b \vee \neg c \rangle$$

All these undercuts say the same thing which is that the set $\{a, b, c\}$ is inconsistent together with the formula $\neg a \vee \neg b \vee \neg c$. As a result, this can be captured by the last undercut listed above. Note this is the maximally conservative undercut amongst the undercuts listed, and moreover it is a canonical undercut. This example therefore illustrates how the canonical undercuts are the undercuts that (in a sense) represent all the other undercuts.

So choosing classical logic as the base logic gives us a wider range of choices for defining attacks. This has advantages if we want to better capture argumentation as arising in natural language, or to more precisely capture counterarguments generated from certain kinds of knowledge. However, it does also mean that we need to be aware of the consequences of our choice of definition for attacks when using a generative approach to instantiating argument graphs (as we will discuss in the next section).

5. Argument graphs

We now investigate the options for instantiating argument graphs. We start with descriptive argument graphs, and then turn to generative argument graphs, using simple logic and classical logic.

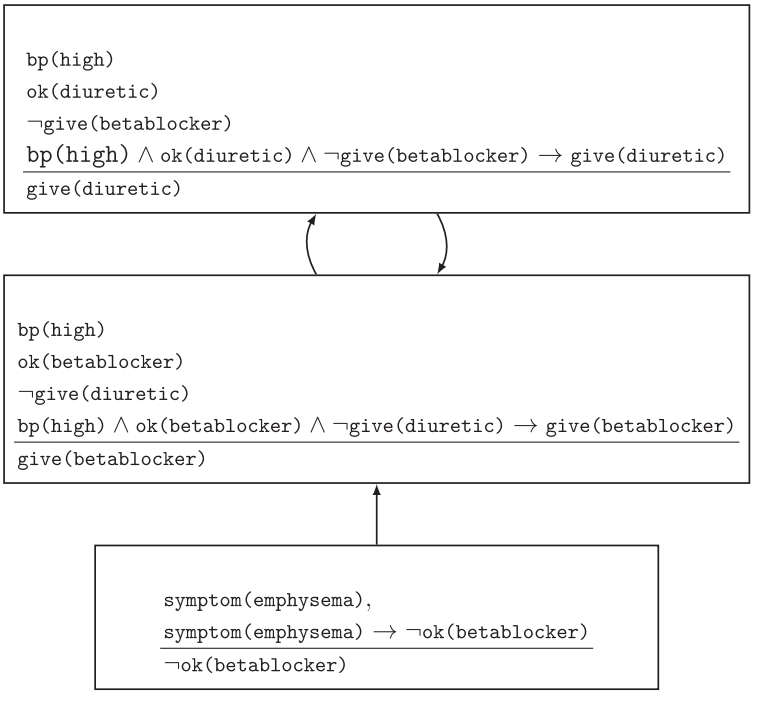
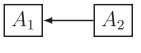


Figure 4. A descriptive graph representation of the abstract argument graph in Figure 1 using simple logic. The atom bp(high) denotes that the patient has high blood pressure. Each attack is a simple undercut by one argument on another. For the first argument, the premises include the assumptions ok(diuretic) and ¬give(betablocker) in order to apply its simple rule. Similarly, for the second argument, the premises include the assumptions ok(betablocker) and ¬give(diuretic) in order to apply its simple rule.

5.1. *Descriptive graphs*

For the descriptive approach to argument graphs, we assume that we have some abstract argument graph as the input, together with some informal description of each argument. For instance, when we listen to a debate on the radio, we may identify a number of arguments and counterarguments, and for each of these we may be able to write a brief text summary. So if we then want to understand this argumentation in more detail, we may choose to instantiate each argument with a deductive argument. So for this task we choose the appropriate logical formulae for the premises and claim for each argument (compatible with the choice of base logic). Examples of descriptive graphs are given in Figure 4 using simple logic, and in Example 5.1 and Figure 5 using classical logic.

EXAMPLE 5.1 Consider the following argument graph where A_1 is "The flight is low cost and luxury, therefore it is a good flight", and A_2 is "A flight cannot be both low cost and luxury".



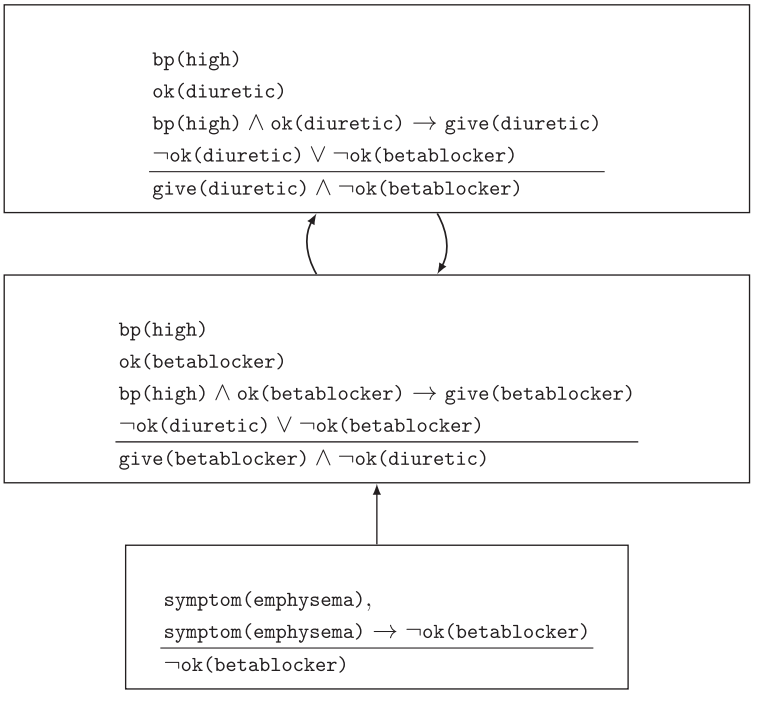
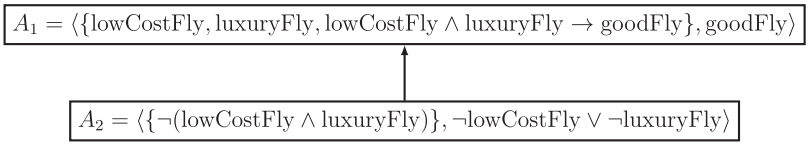


Figure 5. A descriptive graph representation of the abstract argument graph in Figure 1 using classical logic. The atom `bp(high)` denotes that the patient has high blood pressure. The top two arguments rebut each other (i.e. the attack is classical defeating rebut). For this, each argument has an integrity constraint in the premises that says that it is not ok to give both beta blocker and diuretic. So the top argument is attacked on the premise `ok(diuretic)` and the middle argument is attacked on the premise `ok(betablocker)`. So we are using the `ok` predicate as an normality condition for the rule to be applied (as suggested in Section 4.1).

For this, we instantiate the arguments in the above abstract argument graph to give the following descriptive graph. So in the descriptive graph below, A_2 is a classical undercut to A_1.



So for the approach of descriptive graphs, we do not assume that there is an automated process that constructs the graphs. Rather the emphasis is on having a formalisation that is a good representation of the argumentation. This is so that we can formally analyse the descriptive graph, perhaps as part of some sense-making or decision-making process. Nonetheless, we can envisage that in the medium term natural language processing technology will be able to parse the text or speech (for instance in a discussion paper or in a debate) in order to automatically identify the premises and claim of each argument and counterargument.

Since we are primarily interested in representational and analytical issues when we use descriptive graphs, a richer logic such as classical logic is a more appealing formalism than a weaker base logic such as simple logic. Given a set of real-world arguments, it is often easier to model them using deductive arguments with classical logic as the base logic than a "rule-based logic" like

simple logic as the base logic. For instance, in Example 5.1, the undercut does not claim that the flight is not low cost, and it does not claim that it is not luxury. It only claims that the flight cannot be both low cost *and* luxury. It is natural to represent this exclusion using disjunction. As another example of the utility of classical logic as base logic, consider the importance of quantifiers in knowledge which require a richer language such as classical logic for reasoning with them. Moreover, if we consider that many arguments are presented in natural language (spoken or written), and that formalising natural language often calls for a richer formalism such as classical logic (or even richer), it is valuable to harness classical logic in formalisations of deductive argumentation.

5.2. *Generative graphs based on simple logic*

Given a knowledgebase Δ, we can generate an argument graph $G = (\mathcal{A}, \mathcal{R})$, where $\mathcal{A}$ is the set of simple arguments obtained from Δ as follows and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is simple undercut.

DEFINITION 5.2 Let Δ be a simple logic knowledgebase. A *simple exhaustive graph* for Δ is an argument graph $G = (\mathcal{A}, \mathcal{R})$, where $\mathcal{A}$ is $\mathsf{Arguments}_s(\Delta)$ and $\mathcal{R}$ is $\mathsf{Attacks}_s(\Delta)$ defined as follows:

$$\mathsf{Arguments}_s(\Delta) = \{\langle \Phi, \alpha \rangle \mid \Phi \subseteq \Delta \text{ and } \langle \Phi, \alpha \rangle \text{ is a simple argument}\}$$

$$\mathsf{Attacks}_s(\Delta) = \{(A, B) \mid A, B \in \mathsf{Arguments}_s(\Delta) \text{ and } A \text{ is a simple undercut of } B\}$$

This is an exhaustive approach to constructing an argument graph from a knowledgebase since all the simple arguments and all the simple undercuts are in the argument graph. We give an example of such an argument graph in Figure 6.

Simple logic has the property that for any argument graph, there is a knowledgebase that can be used to generate it: let (N, E) be a directed graph (i.e. N is a set of nodes, and E is a set of edges between nodes), then there is a simple logic knowledgebase Δ such that $(\mathsf{Arguments}_s(\Delta), \mathsf{Attacks}_s(\Delta))$ is isomorphic to (N, E). So simple exhaustive graphs are said to be *constructively complete* for graphs.

To show that simple exhaustive graphs are constructively complete for graphs, we can use a coding scheme for the premises so that each argument is based on a single simple rule where the antecedent is a conjunction of one or more positive literals, and each consequent is a negative literal unique to that simple rule (i.e. it is an identifier for that rule and therefore for that argument). If we want one argument to attack another, and the attacking argument has the consequent $\neg\alpha$, then the attacked argument needs to have the positive literal α in the antecedent of its simple rule. The restriction of each rule to only have positive literals as conditions in the antecedent, and a negative literal as its consequent, means that the rules cannot be chained. This ensures that the premises of each argument has only one simple rule. We illustrate this in Figure 6.

Simple exhaustive graphs provide a direct and useful way to instantiate argument graphs. There are various ways the definitions can be adapted, such as defining the attacks to be the union of the simple undercuts and the simple rebuts.

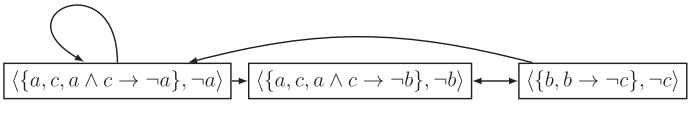


Figure 6. An exhaustive simple logic argument graph where $\Delta = \{a, b, c, a \wedge c \rightarrow \neg a, b \rightarrow \neg c, a \wedge c \rightarrow \neg b\}$. Note that this exhaustive graph contains a self-cycle and an odd length cycle.

5.3. *Generative graphs based on classical logic*

In this section, we consider generative graphs for classical logic. We start with the classical exhaustive graphs which are the same as the simple exhaustive graphs except we use classical arguments and attacks. We show that whilst this provides a comprehensive presentation of the information, its utility is limited for various reasons. We then show that by introducing further information, we can address these shortcomings. To illustrate this, we consider a version of classical exhaustive graphs augmented with preference information. This is just one possibility for introducing extra information into the construction process.

5.3.1. *Classical exhaustive graphs*

Given a knowledgebase Δ, we can generate an argument graph $G = (\mathcal{A}, \mathcal{R})$, where A is the set of arguments obtained from Δ as follows and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is one of the definitions for classical attack.

DEFINITION 5.3 Let Δ be a classical logic knowledgebase. A *classical exhaustive graph* is an argument graph $G = (\mathcal{A}, \mathcal{R})$, where $\mathcal{A}$ is $\mathsf{Arguments}_c(\Delta)$ and $\mathcal{R}$ is $\mathsf{Attacks}_c^X(\Delta))$ defined as follows where X is one of the attacks given in Definition 4.5 such as defeater, direct undercut, or rebuttal.

$$\mathsf{Arguments}_c(\Delta) = \{\langle \Phi, \alpha \rangle \mid \Phi \subseteq \Delta \text{ and } \langle \Phi, \alpha \rangle \text{ is a classical argument}\}$$

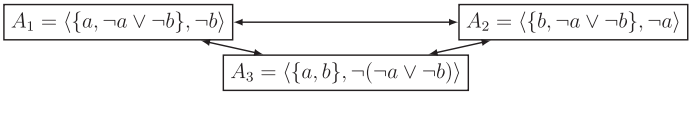
$$\mathsf{Attacks}_c^X(\Delta) = \{(A, B) \in \mathsf{Arguments}_c(\Delta) \times \mathsf{Arguments}_c(\Delta) \mid A \text{ is X of } B\}$$

This is a straightforward approach to constructing an argument graph from a knowledgebase since all the classical arguments and all the attacks (according to the chosen definition of attack) are in the argument graph as illustrated in Figure 7. However, if we use this exhaustive definition, we obtain infinite graphs, even if we use a knowledgebase with few formulae. For instance, if we have an argument $\langle \{a\}, a \rangle$, we also have arguments such as $\langle \{a\}, a \vee a \rangle$, $\langle \{a\}, a \vee a \vee a \rangle$, etc., as well as $\langle \{a\}, \neg\neg a \rangle$, etc.

Even though the graph is infinite, we can present a finite representative of it, by just presenting a representative of each class of structurally equivalent arguments (as considered in Amgoud, Besnard, & Vesic 2011), where we say that two arguments A_i and A_j are *structurally equivalent* in $G = (\mathcal{A}, \mathcal{R})$ when the following conditions are satisfied: (1) if A_k attacks A_i, then A_k attacks A_j; (2) if A_k attacks A_j, then A_k attacks A_i; (3) if A_i attacks A_k, then A_j attacks A_k; and (4) if A_i attacks A_k, then A_j attacks A_k. For example, in Figure 7, the argument A_4 is a representative for $\langle \{b\}, b \rangle$, $\langle \{b\}, a \vee b \rangle$, $\langle \{b\}, \neg a \vee b \rangle$, etc.

We can also ameliorate the complexity of classical exhaustive graphs by presenting a focal graph (as discussed in Section 2). We illustrate this in the following example.

EXAMPLE 5.4 Consider the knowledgebase $\Delta = \{a, b, \neg a \vee \neg b\}$ as used in Figure 7. Suppose we have the focus $\Pi = \{A_1, A_2\}$ (i.e. the arguments of interest), then the focal graph is as follows.



Various postulates have been proposed for classical exhaustive graphs (e.g. Gorogiannis & Hunter 2011). Some of these are concerned with consistency of the set of premises (or set of claims) obtained from the arguments in an extension according to one of Dung's dialectical semantics. We

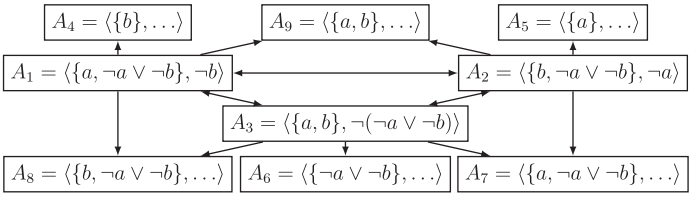


Figure 7. An exhaustive classical logic argument graph where $\Delta = \{a, b, \neg a \vee \neg b\}$ and the attack is direct undercut. Note argument A_4 represents all arguments with a claim that is implied by b, argument A_5 represents all arguments with a claim that is implied by a, argument A_6 represents all arguments with a claim that is implied by $\neg a \vee \neg b$, argument A_7 represents all arguments with a claim that is implied by $a \wedge \neg b$ except $\neg b$ or any claim implied by a or any claim implied by $\neg a \vee \neg b$, argument A_8 represents all arguments with a claim that is implied by $\neg a \wedge b$ except $\neg a$ or any claim implied by b or any claim implied by $\neg a \vee \neg b$, and argument A_9 represents all arguments with a claim that is implied by $a \wedge b$ except $\neg(\neg a \vee \neg b)$ or any claim implied by a or any claim implied by b.

give a useful example of one of these consistency postulates (taken from Gorogiannis & Hunter 2011): for a classical exhaustive graph G, the *consistent extension property* is defined as follows where $\mathsf{Extension}^S(G)$ is the set of extensions obtained from the argument graph G according to the semantics S (e.g. grounded, preferred, etc.).

$$\bigcup_{A \in \Gamma} \mathsf{Support}(A) \not\models \bot, \text{for all } \Gamma \in \mathsf{Extension}^S(G)$$

What has been found is that for some choices of attack relation (e.g. rebuttal), the consistent extension property is not guaranteed (as in Example 5.5), whereas for other choices of attack relation (e.g. direct undercut), the consistent extension property is guaranteed. We illustrate a consistent set of premises obtained from arguments in a preferred extension in Example 5.6.

Example 5.5 Let $\Delta = \{a \wedge b, \neg a \wedge c\}$. For the reviewed semantics for rebut, the following are arguments in any extension: $A_1 = \langle\{a \wedge b\}, b\rangle$ and $A_2 = \langle\{\neg a \wedge c\}, c\rangle$. Clearly $\{a \wedge b, \neg a \wedge c\} \vdash \bot$. Hence, the consistent extension property fails for rebut.

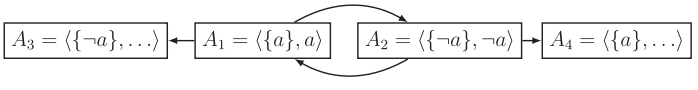
Example 5.6 Consider the argument graph given in Figure 7. There are three preferred extensions $\{A_1, A_5, A_6, A_7\}$, $\{A_2, A_4, A_6, A_8\}$, and $\{A_3, A_4, A_5, A_9\}$. In each case, the union of the premises is consistent. For instance, for the first extension,

$$\mathsf{Support}(A_1) \cup \mathsf{Support}(A_5) \cup \mathsf{Support}(A_6) \cup \mathsf{Support}(A_7) \not\vdash \bot$$

The failure of the consistent extension property with some attack relations is an issue that may be interpreted as a weakness of the attack relation or of the specific semantics, and perhaps raises the need for alternatives to be identified. Another response is that it is not the attack relation and dialectical semantics that should be responsible for ensuring that all the premises used in the winning arguments are consistent together. Rather, it could be argued that checking that the premises used are consistent together should be the responsibility of something external to the defeat relation and dialectical semantics, and so knowing whether the consistent extension property holds or not influences what external mechanisms are required for checking. Furthermore, checking consistency of premises of sets of arguments may be part of the graph construction process. For instance, in García and Simari (2004) proposal for dialectical trees, there are constraints on what arguments can be added to the tree based on consistency with the premises of other arguments in the tree.

In the previous section, we saw that the definition for simple exhaustive graphs (i.e. simple logic as the base logic, with simple undercuts) is constructively complete for graphs. In contrast, the definition for classical exhaustive graphs (i.e. classical logic, with any of the definitions for counterarguments) is not constructively complete for graphs. Since the premises of a classical argument are consistent, by definition, it is not possible for a classical argument to attack itself using the definitions for attack given earlier. But, there are many other graphs for which there is no classical logic knowledgebase that can be used to generate a classical exhaustive graph that is isomorphic to it. To illustrate this failure, we consider in Example 5.7 the problem of constructing a component with two arguments attacking each other. Note this is not a pathological example as there are many graphs that contain a small number of nodes and that cannot be generated as a classical exhaustive graph.

EXAMPLE 5.7 Let $\Delta = \{a, \neg a\}$ be a classical logic knowledgebase. Hence, there are two classical arguments A_1 and A_2 that are direct undercuts of each other. Plus, there is the representative A_3 for arguments with a claim that is strictly weaker than a (i.e. the claim b is such that $\{a\} \vdash b$ and $\{b\} \nvdash \{a\}$), and there is the representative A_4 for arguments with a claim that is strictly weaker than $\neg a$ (i.e. the claim b is such that $\{\neg a\} \vdash b$ and $\{b\} \nvdash \{\neg a\}$).



To conclude our discussion of classical exhaustive graphs, the definitions ensure that all the ways that the knowledge can be used to generate classical arguments and classical counterarguments (modulo the choice of attack relation) are laid out. However, for some choices of attack relation, there is a question of consistency (which may be an issue if no further consistency checking is undertaken). Also, the definition for classical exhaustive graphs is not structurally complete for graphs (which means that many argument graphs cannot be generated as classical exhaustive graphs). Perhaps more problematical is that even for small knowledgebases, the classical exhaustive graphs that are generated are complex. Because of the richness of classical logic, the knowledge can be in different combinations to create many arguments. Whilst we can ameliorate this problem by presenting argument graphs using a representative for a class of structurally equivalent arguments, and by using focal graphs, the graphs can still be large. What is evident from this is that there needs to be more selectivity in the process of generating argument graphs. The generation process needs to discriminate between the arguments (and/or the attacks) based on extra information about the arguments and/or information about the audience. There are many ways that this can be done. In the next section, we consider a simple proposal for augmenting the generation process with preferences over arguments.

5.3.2. *Preferential exhaustive graphs*

One of the first proposals for capturing the idea of preferences in constructing argument graphs was preference-based argumentation frameworks (PAF) by Amgoud & Cayrol (2002). This generalises Dung's definition for an argument graph by introducing a preference relation over arguments that in effect causes an attack to be ignored when the attacked argument is preferred over the attacker. So in PAF, we assume a preference relation over arguments, denoted $\preceq$, as well as a set of arguments $\mathcal{A}$ and an attack relation $\mathcal{R}$. From this, we need to define a defeat relation $\mathcal{D}$ as follows, and

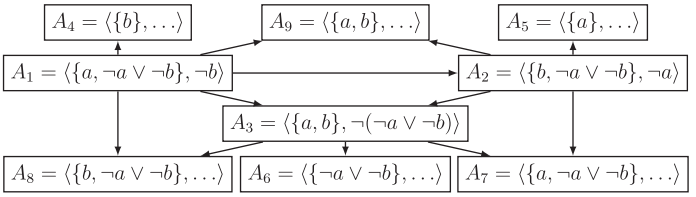


Figure 8. The preferential exhaustive graph where the knowledgebase is $\Delta = \{a, b, \neg a \vee \neg b\}$ and the attack is direct undercut. This is the same knowledgebase and attack relation as in Figure 7. For the preference relation, A_1 is preferred to all other arguments, A_2 is preferred to all other arguments apart from A_1, and the remaining arguments are equally preferred. So for all i such that $i \neq 1$, $A_1 \prec A_i$, and for all i such that $i \neq 1$ and $i \neq 2$, $A_2 \prec A_i$.

then $(\mathcal{A}, \mathcal{D})$ is used as the argument graph, instead of $(\mathcal{A}, \mathcal{R})$, with Dung's usual definitions for extensions.

$$\mathcal{D} = \{(A_i, A_j) \in \mathcal{R} \mid (A_j, A_i) \notin \preceq\}$$

So with this definition for defeat, extensions for a preference-based argument graph $(\mathcal{A}, \mathcal{R}, \preceq)$ can be obtained as follows: for S denoting complete, preferred, stable, or grounded semantics, $\Gamma \subseteq \mathcal{A}$, Γ is an extension of $(\mathcal{A}, \mathcal{R}, \preceq)$ w.r.t. semantics S iff Γ is an extension of $(\mathcal{A}, \mathcal{D})$ w.r.t. semantics S.

We now revise the definition for classical exhaustive graphs to give the following definition for preferential exhaustive graphs.

DEFINITION 5.8 Let Δ be a classical logic knowledgebase. A *preferential exhaustive graph* is an argument graph $(\mathsf{Arguments}_c(\Delta), \mathsf{Attacks}^X_{c,\preceq}(\Delta))$ defined as follows where X is one of the attacks given in Definition 4.5 such as defeater, direct undercut, or rebuttal.

$$\mathsf{Arguments}_c(\Delta) = \{\langle\Phi, \alpha\rangle \mid \Phi \subseteq \Delta \;\&\; \langle\Phi, \alpha\rangle \text{is a classical argument}\}$$

$$\mathsf{Attacks}^X_{c,\preceq}(\Delta) = \{(A, B) \mid A, B \in \mathsf{Arguments}_c(\Delta) \;\&\; A \text{ is X of } B \;\&\; (B, A) \notin \preceq\}$$

We give an illustration of a preferential exhaustive graph in Figure 8, and we give an illustration of a focal graph obtained from a preferential exhaustive graph in Example 5.9.

EXAMPLE 5.9 This example concerns two possible treatments for glaucoma caused by raised pressure in the eye. The first is a prostaglandin analogue (PGA) and the second is a beta blocker (BB). Let Δ contain the following six formulae. The atom $\mathtt{p}_1$ is the fact that the patient has glaucoma, the atom $\mathtt{p}_2$ is the assumption that it is ok to give PGA, and the atom $\mathtt{p}_3$ is the assumption that it is ok to give BB. Each implicational formula (i.e. $\mathtt{p}_4$ and $\mathtt{p}_5$) captures the knowledge that if a patient has glaucoma, and it is ok to give a particular drug, then give that drug. Formula $\mathtt{p}_6$ is an integrity constraint that ensures that only one treatment is given for the condition.

$$\mathtt{p}_1 = \mathtt{glaucoma} \qquad \mathtt{p}_4 = \mathtt{glaucoma} \wedge \mathtt{ok(PGA)} \rightarrow \mathtt{give(PGA)}$$

$$\mathtt{p}_2 = \mathtt{ok(PGA)} \qquad \mathtt{p}_5 = \mathtt{glaucoma} \wedge \mathtt{ok(BB)} \rightarrow \mathtt{give(BB)}$$

$$\mathtt{p}_3 = \mathtt{ok(BB)} \qquad \mathtt{p}_6 = \neg\mathtt{ok(PGA)} \vee \neg\mathtt{ok(BB)}$$

There are numerous arguments that can be constructed from this set of formulae such as the following.

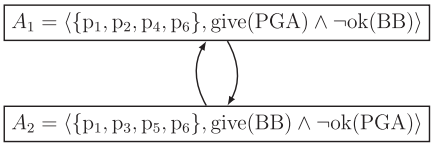
$$A_1 = \langle\{\mathtt{p}_1, \mathtt{p}_2, \mathtt{p}_4, \mathtt{p}_6\}, \mathtt{give(PGA)} \wedge \neg\mathtt{ok(BB)}\rangle \qquad \mathtt{A}_5 = \langle\{\mathtt{p}_1, \mathtt{p}_2, \mathtt{p}_4\}, \mathtt{give(PGA)}\rangle$$

$$A_2 = \langle\{\mathtt{p}_1, \mathtt{p}_3, \mathtt{p}_5, \mathtt{p}_6\}, \mathtt{give(BB)} \wedge \neg\mathtt{ok(PGA)}\rangle \qquad \mathtt{A}_6 = \langle\{\mathtt{p}_1, \mathtt{p}_3, \mathtt{p}_5\}, \mathtt{give(BB)}\rangle$$

$$A_3 = \langle\{\mathtt{p}_2, \mathtt{p}_3\}, \mathtt{ok(PGA)} \wedge \mathtt{ok(BB)}\rangle \qquad \mathtt{A}_7 = \langle\{\mathtt{p}_2, \mathtt{p}_6\}, \neg\mathtt{ok(BB)}\rangle$$

$$A_4 = \langle\{\mathtt{p}_6\}, \neg\mathtt{ok(PGA)} \vee \neg\mathtt{ok(BB)}\rangle \qquad \mathtt{A}_8 = \langle\{\mathtt{p}_3, \mathtt{p}_6\}, \neg\mathtt{ok(PGA)}\rangle$$

Let $\mathsf{Arguments}_c(\Delta)$ be the set of all classical arguments that can be constructed from Δ, and let the preference relation $\preceq$ be such that $A_i \preceq A_1$ and $A_i \preceq A_2$ for all i such that $i \neq 1$ and $i \neq 2$. Furthermore, let $\Pi = \{A_1, A_2\}$ be the focus (i.e. the arguments of interest). In other words, we know that each of these two arguments in the focus contains all the information we are interested in (i.e. we want to determine the options for treatment taking into account the integrity constraint). This would give us the following focal graph.



By taking this focal graph, we have ignored arguments such as A_3 to A_8 which do not affect the dialectical status of A_1 or A_2 given this preference relation.

Using preferences is a general approach. There is no restriction on what preference relation we use over arguments, and there are various natural interpretations for this ranking such as capturing belief for arguments (where the belief in the argument can be based on the belief for the premises and/or claim), capturing the relative number of votes for arguments (where a group of voters will vote for or against each argument), etc.

To conclude, by introducing preferences over arguments, we can reduce the number of attacks that occur. Using preferences over arguments is a form of meta-information, and with the definition for preference-based argumentation (as defined by Amgoud & Cayrol 2002), it supports selectivity in generating argument graphs that discriminates between arguments and thereby between attacks. With this definition more practical argument graphs can be constructed than with the definition for classical exhaustive graphs. Furthermore, using an appropriate definition for the preference relation, the definition for preference exhaustive graphs is structurally complete for graphs, and for some choices of preference relation and dialectical semantics, the consistent extension property holds (see, for example, the use of probability theory for obtaining preferences over arguments by Hunter 2013).

6. Comparison with other approaches

To summarise what we have covered in this tutorial, the use of deductive arguments gives a simple, transparent, and precise way of representing arguments and counterarguments. Furthermore, by appropriate choice of base logic, it can capture a wide variety of real-world examples in descriptive graphs, and by augmenting the information about the arguments (e.g. by using preferences or probabilistic information), practical generative graphs can be obtained.

We now briefly compare the approach with the other approaches to structured argumentation considered in this special issue.

6.1. *Assumption-based argumentation*

This is a general framework for defining logic-based argumentation systems. As with deductive argumentation, a language needs to be specified for representing the information. Rule-based languages and classical logic are possibilities. In addition, a set of inference rules needs to be specified. This corresponds to a base logic consequence relation augmented with a set of domain-specific inference rules. A subset of the language is specified as the set of assumptions, and each argument is a subset of these assumptions. The assumptions in an argument can be viewed as the premises of the argument, with the claim being implicit. The approach incorporates a generalised notion of negation that specifies which individual formulae contradict an assumption, and a form of undercut is defined using this notion. Given a knowledgebase, all the arguments and counterarguments can be generated, and the attacks relation identified, thereby producing an "exhaustive graph". However, a significant feature of the approach is a set of proof procedures for determining whether there exists an argument with a given claim in a preferred, grounded, or ideal extension. Many of the examples used to illustrate and develop assumption-based argumentation (ABA) are a simple rule-based language (analogous to simple logic for deductive argumentation). However, the approach is general and can use a wide variety of underlying logics including classical logic (though the same issues as raised in Section 5.3.1 would arise for ABA).

6.2. *ASPIC+*

This is a general framework for defining structured argumentation systems. As with deductive argumentation and ABA, a language needs to be specified for representing the information. Rule-based languages and classical logic are possibilities. As with ABA, a set of inference rules needs to be specified. Inference rules can be domain specific. This contrasts with deductive argumentation where inference rules are specified by the base logic. An argument in ASPIC+ contains the structure of the proof used to obtain the claim from the premises, and so the same pair of premises and claim can potentially result in many arguments. The defeasibility of an argument based on a defeasible rule can come from a counterargument that attacks the use of defeasible rule. This is done via a naming mechanism for the rules, so an attacker of a defeasible rule has a claim that is the contrary (negation) of the name of that rule. The defeasibility of a defeasible rule can also come from counterarguments that either rebut or undercut (called undermine in ASPIC+) an argument. ASPIC+ incorporates a preference relation over arguments which allows for some attacks to be ignored. Given a knowledgebase, all the arguments and counterarguments can be generated, and the attacks relation identified, thereby producing an "exhaustive graph". A main objective in the development of ASPIC+ is to identify conditions under which instantiations of ASPIC+ satisfy logical consistency and closure properties.

6.3. *Defeasible logic programming*

This is an approach to argumentation based on a logic programming language. Each argument is generated from the knowledgebase, which contains defeasible rules and strict rules, and these are used to form a tree of arguments and counterarguments: an argument for a claim of interest is used for the root, and then counterarguments to this root are children to this root, and then by recursion for each counterargument, the counterarguments to it are given as its children. The construction process imposes some consistency constraints and ensures that no branch is infinite. This tree provides an exploration of the possible reasons for and against the root of the tree being a warranted argument. To compare the approach with that of deductive argumentation, each argument in defeasible logic programming (DeLP) can be viewed as a deductive argument where the base logic is a form of logic programming consequence relation, and each counterargument can

be seen as a form of undercut, but the approach does not involve instantiating abstract argument graphs. A number of variants of DeLP have been proposed incorporating features such as variable strength arguments (Martinez, Garcia, & Simari 2008) and possibility theory (Alsinet, Chesñevar, Godo, & Simari 2008).

7. Further reading

We provide further reading on formalisation of deductive arguments and counterarguments, properties of exhaustive graphs, the importance of selectivity in generating argument graphs, and on automated reasoning.

7.1. *Deductive arguments and counterarguments*

There have been a number of proposals for deductive arguments using classical propositional logic (Amgoud & Cayrol 2002, Besnard & Hunter 2001, Cayrol 1995, Gorogiannis & Hunter 2011), classical predicate logic (Besnard & Hunter 2005), description logic (Black, Hunter, & Pan 2009; Moguillansky, Wassermann, & Falappa 2010; Zhang & Lin 2013; Zhang, Zhang, Xu, & Lin 2010), temporal logic (Mann & Hunter 2008), simple (defeasible) logic (Governatori, Maher, Antoniou, & Billington 2004; Hunter 2010), conditional logic (Besnard, Gregoire, & Raddaoui 2013), and probabilistic logic (Haenni 1998, Haenni 2001, Hunter 2013).

There has also been progress in understanding the nature of classical logic in computational models of argument. Various types of counterarguments have been proposed including rebuttals (Pollock 1987, Pollock 1992), direct undercuts (Cayrol 1995; Elvang-Gøransson & Hunter 1995; Elvang-Gøransson, Krause, & Fox 1993), and undercuts and canonical undercuts (Besnard & Hunter 2001). In most proposals for deductive argumentation, an argument A is a counterargument to an argument B when the claim of A is inconsistent with the support of B. It is possible to generalise this with alternative notions of counterargument. For instance, with some common description logics, there is not an explicit negation symbol. In the proposal for argumentation with description logics, Black et al. (2009) used the description logic notion of *incoherence* to define the notion of counterargument: a set of formulae in a description logic is incoherent when there is no set of assertions (i.e. ground literals) that would be consistent with the formulae. Using this, an argument A is a counterargument to an argument B when the claim of A together with the support of B is incoherent.

Meta-arguments for deductive argumentation was first proposed by Wooldridge, McBurney, & Parsons (2005), and the investigation of the representation of argument schemes in deductive argumentation was first proposed by Hunter (2008).

7.2. *Properties of exhaustive argument graphs*

In order to investigate how Dung's notion of abstract argumentation can be instantiated with classical logic, Cayrol (1995) presents results concerning stable extensions of argument graphs where the nodes are classical logic arguments, and the attacks are direct undercuts. As well as being the first paper to propose instantiating abstract argument graphs with classical arguments, it also showed how the premises in the arguments in the stable extension correspond to maximal consistent subsets of the knowledgebase, when the attack relation is direct undercut.

Insights into the options for instantiating abstract argumentation with classical logic can be based on postulates. Amgoud & Besnard (2009) have proposed a consistency condition and they examine special cases of knowledgebases and symmetric attack relations and whether consistency is satisfied in this context. Then Amgoud & Besnard (2010) extend this analysis by showing

correspondences between the maximal consistent subsets of a knowledgebase and the maximal conflict-free sets of arguments.

Given the wide range of options for attack in classical logic, Gorogiannis & Hunter (2011) propose a series of desirable properties of attack relations to classify and characterise attack relations for classical logic. Furthermore, they present postulates regarding the logical content of extensions of argument graphs that may be constructed with classical logic, and a systematic study is presented of the status of these postulates in the context of the various combinations of attack relations and extension semantics.

Use of the notion of generative graphs then raises the question of whether for a specific logical argument system S, and for any graph G, there is a knowledgebase such that S generates G. If it holds, then it can be described as a kind of "structural"property of the system (Hunter & Woltran 2013). If it fails then, it means that there are situations that cannot be captured by the system. The approach of simple exhaustive graphs is constructively complete for graphs, whereas the approach of classical exhaustive graphs is not.

Preferences have been introduced into classical logic argumentation and used to instantiate abstract argumentation with preferences by Amgoud & Cayrol (2002). Amgoud and Vesic have shown how preferences can be defined so as to equate inconsistency handling in argumentation with inconsistency handling using Brewka's preferred sub-theories (Amgoud & Vesic 2010).

7.3. *Importance of selectivity in deductive argumentation*

Some of the issues raised with classical exhaustive graphs (i.e. the lack of structural completeness, the failure of consistent extension property for some choices of attack relation, and the correspondences with maximally consistent subsets of the knowledgebase) suggest that often we need a more sophisticated way of constructing argument graphs. In other words, to reflect any abstract argument graph in a logical argument system based on a richer logic, we need to be selective in the choice of arguments and counterarguments from those that can be generated from the knowledgebase. Furthermore, this is not just for theoretical interest. Practical argumentation often seems to use richer logics such as classical logic, and often the arguments and counterarguments considered are not exhaustive. Therefore, we need to better understand how the arguments are selected. For example, suppose agent 1 posits $A_1 = \langle\{b, b \rightarrow a\}, a\rangle$, and agent 2 then posits $A_2 = \langle\{c, c \rightarrow \neg b\}, \neg b\rangle$. It would be reasonable for this dialogue to stop at this point even though there are further arguments that can be constructed from the public knowledge such as $A_3 = \langle\{b, c \rightarrow \neg b\}, \neg c\rangle$. So in terms of constructing the constellation of arguments and counterarguments from the knowledge, we need to know what are the underlying principles for selecting arguments.

Selectivity in argumentation is an important and as yet under-developed topic (Besnard & Hunter 2008). Two key dimensions are selectivity based on object-level information and selectivity based on meta-level information.

- *Selectivity based on object-level information.* In argumentation, object-level information is the information in the premises and claims of the arguments. So if these are generated by deductive reasoning from a knowledgebase, then the object-level information is the information in the knowledgebase. Selectivity based on object-level information is concerned with having a more concise presentation of arguments and counterarguments in an argument graph without changing the outcome of the argumentation. For instance, a more concise presentation can be obtained by removing structurally equivalent arguments or by using focal graphs (as discussed in Section 5.3.1).
- *Selectivity based on meta-level information.* In argumentation, meta-level information is the information about the arguments and counterarguments (e.g. certainty and sources

of the premises in arguments) and information about the participants or audience of the argumentation (e.g. the goals, beliefs, or biases of the audience). Selectivity based on meta-level information is concerned with generating an argument graph using the meta-level information according to sound principles. By using this extra information, a different argument graph may be obtained than would be obtained without the extra information. For instance, with a preference relation over arguments which is a form of meta-level information, preference-based argumentation offers a principled way of generating an argument graph that has potentially fewer attacks between arguments than obtained with the classical exhaustive argument graph (as discussed in Section 5.3.2).

Various kinds of meta-level information can be considered for argumentation including preferences over arguments, weights on arguments, weights on attacks, a probability distribution over models of the language of the deductive argumentation, etc. The need for meta-level information also calls for better modelling of the audience, of what they believe, of what they regard as important for their own goals, etc., is an important feature of selectivity (see, for example, (Hunter 2004a, 2004b), Hunter 2004b). Consider a journalist writing a magazine article on current affairs. There are many arguments and counterarguments that could be included, but the writer is selective. Selectivity may be based on what the likely reader already believes and what they may find interesting. Or, consider a lawyer in court, again there may be many arguments and counterarguments that could be used, but only some will be used. Selection will in part be based on what could be believed by the jury and convince them to take the side of that lawyer. Or, consider a politician giving a speech to an audience of potential voters. Here, the politician will select arguments based on what will be of more interest to the audience. For instance, if the audience is composed of older citizens, there may be more arguments concerning healthcare, whereas if the audience is composed of younger citizens, there may be more arguments concerning job opportunities. So whilst selectivity is clearly important in real-world argumentation, we need principled ways of bring selectivity into structured argumentation such as that based on deductive argumentation.

7.4. *Automated reasoning for deductive argumentation*

For argumentation, it is computationally challenging to generate arguments from a knowledgebase with the minimality constraints using classical logic. 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). The difficult nature of argumentation has been underlined by studies concerning the complexity of finding individual arguments Parsons, Wooldridge, & Amgoud (2003), the complexity of some decision problems concerning the instantiation of argument graphs with classical logic arguments and the direct undercut attack relation Wooldridge, Dunne, & Parsons (2006), and the complexity of finding argument trees Hirsch & Gorogiannis (2009). Encodation of these tasks as quantified Boolean formulae also indicates that development of algorithms is a difficult challenge (Besnard, Hunter, & Woltran 2009), and Post's framework has been used to give a breakdown of where complexity lies in logic-based argumentation (Creignou, Schmidt, Thomas, & Woltran 2011).

Despite the computational complexity results, there has been progress in developing algorithms for constructing arguments and counterarguments. One approach has been to adapt the idea of connection graphs to enable us to find arguments. A connection graph (Kowalski 1975), (Kowalski 1979) is a graph where a clause is represented by a node and an arc (ϕ, ψ) denotes that there is a disjunct in ϕ with its complement being a disjunct in ψ. Essentially this graph is manipulated to obtain a proof by contradiction. Furthermore, finding this set of formulae can substantially reduce

the number of formulae that need to be considered for finding proofs for a claim, and therefore for finding arguments and canonical undercuts. Versions for full propositional logic, and for a subset of first-order logic, have been developed and implemented (Efstathiou & Hunter 2011).

Another approach for algorithms for generating arguments and counterarguments (canonical undercuts) have been given in a proposal that is based on an SAT solver (Besnard, Gregoire, Piette, & Raddaoui 2010). This approach is based on standard SAT technology and it is also based on finding proofs by contradiction.

Acknowledgements

The authors are very grateful to Henry Prakken for some very interesting and valuable discussions on the nature of deductive argumentation. The second author is also grateful to Schloss Dagstuhl, and the organisers and participants of the seminar on Belief Change and Argumentation in Multi-Agent Scenarios (Seminar 13231), for giving him the opportunity to discuss some of the ideas in this tutorial.

References

Alsinet, T., Chesñevar, C., Godo, L., & Simari, G. (2008). A logic programming framework for possibilistic argumentation: Formalization and logical properties. *Fuzzy Sets and Systems, 159*, 1208–1228.

Amgoud, L., & Besnard, Ph. (2009). Bridging the gap between abstract argumentation systems and logic. In L. Godo and A. Pugliese (Eds.), *Proceedings of the third international conference on Scalable Uncertainty Management (SUM'09)* (pp. 12–27). Vol. 5785 of Lecture Notes in Computer Science. Springer.

Amgoud, L., & Besnard, Ph. (2010). A formal analysis of logic-based argumentation systems. In A. Deshpande & A. Hunter (Eds.), *Proceedings of the fourth international conference on Scalable Uncertainty Management (SUM'10)* (pp. 42–55). Vol. 6379 of Lecture Notes in Computer Science. Springer.

Amgoud, L., Besnard, P., & Vesic, S. (2011). Identifying the core of logic-based argumentation systems. In *IEEE 23rd International Conference on Tools with Artificial Intelligence, (ICTAI'11)* (pp. 633–636). IEEE.

Amgoud, L., & Cayrol, C. (2002). A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence, 34*, 197–215.

Amgoud, L., & Vesic, S. (2010). Handling inconsistency with preference-based argumentation. In A. Deshpande & A. Hunter (Eds.), *Proceedings of the 4th international conference on Scalable Uncertainty Management (SUM'10)* (pp. 56–69). Vol. 6379 of Lecture Notes in Computer Science. Springer.

Baroni, P., & Giacomin, M. (2007). On principle-based evaluation of extension-based argumentation semantics. *Artificial Intelligence, 171*, 675–700.

Besnard, P., Gregoire, E., Piette, C., & Raddaoui, B. (2010). Mus-based generation of arguments and counter-arguments. In *Proceedings of the 11th IEEE international conference on Information Reuse and Integration (IRI'10)* (pp. 239–244). IEEE Press.

Besnard, P., Gregoire, E., & Raddaoui, B. (2013). A conditional logic-based argumentation framework. In W. Liu, V. Subrahmanian, & J. Wijsen (Eds.), *Proceedings of the 7th international conference on Scalable Uncertainty Management (SUM'13)* (pp. 44–56). Vol. 7958 of Lecture Notes in Computer Science. Springer.

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

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

Besnard, P., & Hunter, A. (2008). *Elements of argumentation*. MIT Press.

Besnard, P., Hunter, A., & Woltran, S. (2009). Encoding deductive argumentation in quantified Boolean formulae. *Artificial Intelligence, 173*, 1406–1423.

Black, E., Hunter, A., & Pan, J. (2009). An argument-based approach to using multiple ontologies. In *Third international conference on Scalable Uncertainty Management (SUM'09)* (pp. 68–79). Vol. 5785 of Lecture Notes in Computer Science. Springer.

Cayrol, C. (1995). On the relation between argumentation and non-monotonic coherence-based entailment. In *Proceedings of the fourteenth International Joint Conference on Artificial Intelligence (IJCAI'95)* Morgan Kaufmann, (pp. 1443–1448).

Creignou, N., Schmidt, J., Thomas, M., & Woltran, S. (2011). Complexity of logic-based argumentation in Post's framework. *Argument & Computation, 2*, 107–129.

Dung, P. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming, and n-person games. *Artificial Intelligence, 77*, 321–357.

Efstathiou, V., & Hunter, A. (2011). Algorithms for generating arguments and counterarguments in propositional logic. *International Journal of Approximate Reasoning, 52*, 672–704.

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

Elvang-Gøransson, M., & Hunter, A. (1995). Argumentative logics: Reasoning with classically inconsistent information. *Data & Knowledge Engineering, 16*, 125–145.

Elvang-Gøransson, M., Krause, P., & Fox, J. (1993). Acceptability of arguments as "logical uncertainty". In M. Clarke, R. Kruse, & S. Moral (Eds.), *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU'93)* (pp. 85–90). Vol. 747 of Lecture Notes in Computer Science. Springer.

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

Gorogiannis, N., & Hunter, A. (2011). Instantiating abstract argumentation with classical logic arguments: Postulates and properties. *Artificial Intelligence, 175*, 1479–1497.

Governatori, G., Maher, M., Antoniou, G., & Billington, D. (2004). Argumentation semantics for defeasible logic. *Journal of Logic and Computation, 14*, 675–702.

Haenni, R. (1998). Modelling uncertainty with propositional assumptions-based systems. In *Applications of uncertainty formalisms* (pp. 446–470). Vol. 1455 of Lecture Notes in Computer Science. Springer.

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

Hirsch, R., & Gorogiannis, N. (2009). The complexity of the warranted formula problem in propositional argumentation. *Journal of Logic and Computation, 20*, 481–499.

Hunter, A. (2004a). Towards higher impact argumentation. In *Proceedings of the 19th national conference on Artificial Intelligence (AAAI 2004)* (pp. 275–280). MIT Press.

Hunter, A. (2004b). Making argumentation more believable. In *Proceedings of the 19th national conference on Artificial Intelligence (AAAI 2004)* (pp. 269–274). MIT Press.

Hunter, A. (2008). Reasoning about the appropriateness of proponents for arguments. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI'08)* (pp. 89–94). MIT Press.

Hunter, A. (2010). Base logics in argumentation. In P. Baroni, F. Cerutti, M. Giacomin, & G. Simari (Eds.), *Proceedings of the 3rd conference on Computational Models of Argument (COMMA'10)* (pp. 275–286). Vol. 216 of Frontiers in Artificial Intelligence and Applications. IOS Press.

Hunter, A. (2013). A probabilistic approach to modelling uncertain logical arguments. *International Journal of Approximate Reasoning*, *54*, 47–81.

Hunter, A., & Woltran, S. (2013) Structural properties for deductive argument systems. In L. van der Gaag (Ed.), *Proceedings of the 12th European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECSQARU'13)* (pp. 278–289). Vol. 7958 of Lecture Notes in Computer Science. Springer.

Kowalski, R. (1975). A proof procedure using connection graphs. *Journal of the ACM, 22*, 572–595.

Kowalski, R. (1979). *Logic for problem solving*. North-Holland Publishing.

Liao, B., Jin, L., & Koons, R. (2011). Dynamics of argumentation systems: A division-based method. *Artificial Intelligence, 175*, 1790–1814.

Mann, N., & Hunter, A. (2008). Argumentation using temporal knowledge. In P. Besnard, S. Doutre, & A. Hunter (Eds.), *Proceedings of the 2nd conference on Computational Models of Argument (COMMA'08)* (pp. 204–215). Vol. 172 of Frontiers in Artificial Intelligence and Applications. IOS Press.

Martinez, D., Garcia, A., & Simari, G. (2008). An abstract argumentation framework with varied-strength attacks. In G. Brewka and and J. Lang (Eds.), *Proceedings of the 11th international conference on Principles of Knowledge Representation and Reasoning (KR'08)* (pp. 135–144). AAAI Press.

McCarthy, J. (1980). Circumscription: A form of non-monotonic reasoning. *Artificial Intelligence, 13*, 23–79.

Moguillansky, M., Wassermann, R., & Falappa, M. (2010). *An argumentation machinery to reason over inconsistent ontologies* (Vol. 6433, pp. 100–109). Springer.

Parsons, S., Wooldridge, M., & Amgoud, L. (2003). Properties and complexity of some formal inter-agent dialogues. *Journal of Logic and Computation, 13*, 347–376.

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

Pollock, J. (1992). How to reason defeasibly. *Artificial Intelligence, 57*, 1–42.

Wooldridge, M., Dunne, P., & Parsons, S. (2006). On the complexity of linking deductive and abstract argument systems. In *Proceedings of the 21st national conference on Artificial Intelligence (AAAI'06)* (pp. 299–304). AAAI Press.

Wooldridge, M., McBurney, P., & Parsons, S. (2005). On the meta-logic of arguments. In *Argumentation in multi-agent systems* (pp. 42–56). Vol. 4049 of Lecture Notes in Computer Science. Springer.

Zhang, X., & Lin, Z. (2013). An argumentation framework for description logic ontology reasoning and management. *Journal of Intelligent Information Systems, 40*, 375–403.

Zhang, X., Zhang, Z., Xu, D., & Lin, Z. (2010). *Argumentation-based reasoning with inconsistent knowledge bases* (Vol. 6085, pp. 87–99). Springer.