Log. Univers. 5 (2011), 255–318

1661-8297/11/020255-64, *published online* October 14, 2011
DOI 10.1007/s11787-011-0036-3

Logica Universalis

Dung's Argumentation is Essentially Equivalent to Classical Propositional Logic with the Peirce–Quine Dagger

Dov M. Gabbay

Abstract. In this paper we show that some versions of Dung's abstract argumentation frames are equivalent to classical propositional logic. In fact, Dung's attack relation is none other than the generalised Peirce–Quine dagger connective of classical logic which can generate the other connectives $\neg, \wedge, \vee, \rightarrow$ of classical logic. After establishing the above correspondence we offer variations of the Dung argumentation frames in parallel to variations of classical logic, such as resource logics, predicate logic, etc., etc., and create resource argumentation frames, predicate argumentation frames, etc., etc. We also offer the notion of logic proof as a geometrical walk along the nodes of a Dung network and thus we are able to offer a geometrical abstraction of the notion of inference based argumentation. Thus our paper is also a contribution to the question:

"What is a logical system"

in as much as it integrates logic with abstract argumentation networks.

Mathematics Subject Classification (2010). 03B05, 03B22, 03B47, 03B99.

Keywords. Argumentation theory, Peirce–Quine dagger, predicate argumentation, Boolean networks, resource based argumentation.

1. Classical Logic as a Network

The primary aim of this paper is to integrate logic with Dung's abstract argumentation networks. We get as a by-product of this effort various generalisations of Dung's network as well as applications to some problems in the area of argumentation.

So the paper is about argumentation but the motivation behind the particular steps taken is integration with logic. The local researcher, immersed in

Research done under ISF project "Integrating Logic and Network Reasoning".

practical argumentation theory, may perceive some of our moves as possibly unnecessary pure mathematics. See, however, the discussion in Sect. 2.5.

In the past 40 years logic has undergone a serious evolutionary development. The meteoric rise of the applied areas of computer science and artificial intelligence put pressure on traditional logic to evolve. There was the urgent need to develop new logics in order to provide better models of human behaviour and actions. Such models are used to help design products which aid/replace the human in his daily activity. As a result, a rich variety of new logics have been developed and there was the need for a new unifying methodology for the chaotic landscape of the new logics.

Thus the question of

"what is a logical system"

is repeatedly being asked and answered, by myself as well as other colleagues.

An important step in this search is the problem of integration of general network reasoning (Bayesian nets, Neural nets, Argumentation nets, Inheritance nets, Ecological nets, etc., etc.) with discrete logical systems of reasoning (classical logics, defeasible logics, logic programming, temporal and modal logics, etc., etc.).

We are going to present classical logic in a certain way, to make it most compatible with argumentation networks. This is done in Sect. 1.1, where we formulate classical logic with a special unary connective which we call the Peirce–Quine–Dung dagger.

Section 1.2 gives semantic tableaux for our logic and in Sect. 1.3 we axiomatise the consequence relation of this logic. Later in Sect. 2 we shall present certain restricted version of Dung's networks which correspond exactly to our logic. We then map argumentation concepts to logical concepts. For example in Sect. 2.3 we show that semantic tableaux correspond to admissible sets in argumentation.

1.1. Peirce's Arrow, Quine's Dagger and the Sheffer Stroke

As a first step towards showing the equivalence of classical logic with abstract argumentation frames we present the above connectives for classical logic.

Classical propositional logic is traditionally formulated with a set of atomic propositions and some or all of the connectives below:

- $1 =$ truth $= \top$
- $0 =$ falsity $= \bot$
- $x \wedge y = \min(x, y)$
- $x \vee y = \max(x, y)$
- $\neg x = 1 - x$
- $x \to y = \max(1 - x, y)$.

Other connectives were put forward for the classical propositional calculus, among them the Sheffer stroke $x \uparrow y$ introduced in 1913 [37], and Peirce's arrow $x \downarrow y$, discovered in 1880 and published thirty years later [33]. Peirce's arrow was also discovered by Quine under the name Quine's dagger, see report in [27]. These connectives were introduced for logical methodological reasons in the foundations of classical logic. They turned up to be also very useful

Table 1.

A	B	$A \uparrow B$	$A \downarrow B$
0	1	1	0
0	0	1	1
1	1	0	0
1	0	1	0

as gates in circuit design, as the NAND and NOR gates. So, in fact, these connectives are as fundamental as the usual ones.

The table for these connectives is

$$x \uparrow y = \max(1 - x, 1 - y)$$
$$x \downarrow y = 1 - \max(x, y).$$

The following (in Table 1) is the truth table for these connectives:

The Sheffer stroke is equivalent to

$$A \uparrow B \equiv \neg A \vee \neg B$$

and the Peirce–Quine dagger is equivalent to

$$A \downarrow B \equiv \neg A \wedge \neg B.$$

The ordinary Boolean connectives are definable from these unary connectives as follows.

- $\neg A \equiv A \uparrow A \equiv A \downarrow A$
- $A \wedge B \equiv \neg(A \uparrow B) \equiv (\neg A) \downarrow (\neg B)$
- $A \vee B \equiv (\neg A) \uparrow (\neg B) \equiv \neg(A \downarrow B)$.

There exist axiomatisations of classical propositional logic in terms of the Sheffer stroke as the only connective, see [35,36], and I assume one can do the same for the case of the Peirce–Quine dagger. The two unary connectives are interdefinable:

- $A \downarrow B \equiv \neg((\neg A) \uparrow (\neg B))$
- $A \uparrow B \equiv \neg((\neg A) \downarrow (\neg B))$

We now formulate classical logic in a certain way, ready to be turned into a Dung network.

Let Δ be a finite multiset of wffs, $\Delta = \{A_1, \ldots, A_n\}$.[1] Define the connective $\Downarrow \Delta$ as follows:

- $\Downarrow \Delta = 1$ iff $\bigwedge_{i=1}^{n} A_i = 0$

Thus

- $\Downarrow \Delta = \bigwedge_{i=1}^{n} \neg A_i = \bigwedge_{A \in \Delta} \neg A$

  Thus the truth function for $\Downarrow \{x_1, \ldots, x_n\}$ is[2]
- $\Downarrow (x_1, \ldots, x_n) = 1 - \max(x_i)$.

[1] We use $\{\ldots\}$ to denote multisets.

[2] Note that this truth function is meaningful for three values as well, i.e. if x_i range over $\{0, 1/2, 1\}$.

We have that the traditional connectives $\neg$ and $\wedge$ are definable

- $\neg A \equiv \Downarrow \{A\}$
- $\bigwedge_{i=1}^{n} A_i \equiv \Downarrow \{\Downarrow \{A_i\} | i = 1, \ldots, n\}$
- $\bigvee_{i=1}^{n} A_i \equiv \Downarrow\Downarrow \{A_1, \ldots, A_n\}$.

In fact it might be easier to preset an axiom system for $\Downarrow$ than for $\downarrow$.

Let us call this connective the Peirce–Quine–Dung Dagger. This name is appropriate because as we shall see in the next section, the Dung attack relation (done by a dagger) corresponds to this connective!

We shall refer to $\Downarrow$ by the short name "Dagger".

This connective is definable using $\downarrow$ because $\wedge$ and $\neg$ are definable from $\downarrow$, as we have seen. It generalises $\downarrow$ because $\Downarrow \{A, B\} = A \downarrow B$.

Definition 1.1 (*Dwffs: wffs with the Dagger connective*). A Dwff is defined by induction:

1. Any atomic q is a Dwff of level 0.
2. If $\Delta = \{A_1, \ldots, A_n\}$ is a multiset of Dwffs then $A = \Downarrow \Delta$ is also a Dwff. A_i are said to be immediate subformulas of A. The level of A is $1 + \max\{\text{level of } A_i\}$.

Definition 1.2 (*Trees*). A system (S, R, t) is a finite tree iff the following holds:

1. $R \subseteq S \times S$. t is an element of S, and S is finite.
2. For any $x \in S, x \neq t$, there exists a unique y such that yRx. y is called the predecessor of x.
3. For every x there exists a unique sequence $(t, x_1, \ldots, x_n = x)$ such that $tRx_1 \wedge x_1Rx_2 \wedge \ldots \wedge x_{n-1}Rx_n$.

For $x = t$ we have (t) as the sequence.

Definition 1.3 (*Construction tree for a D-formula*). Let A be a Dwff. Then a tuple (S, R, t, α) is a construction tree for A iff (S, R, t) is a tree and α is a function associating a Dwff $\alpha(x)$ with each $x \in S$ such that the following holds.

1. $\alpha(x)$ is atomic if x is an endpoint (i.e. $\neg\exists y(xRy)$).
2. For all $x \in S$ which are predecessors, we have
$$\alpha(x) = \Downarrow \{\alpha(y) | xRy\}.$$
3. $\alpha(t) = A$.

Definition 1.4 (*Acyclic decorated ordering*). A system (S, R, α) is said to be an acyclic decorated ordering iff the following holds:

1. (S, R) is a finitary acyclic ordering. This means that $R \subseteq S \times S$ is a binary relation such that for all $x, \{y | xRy\}$ is a finite set and such that for no $x_1, \ldots, x_m$ do we have $x_1Rx_2 \wedge x_2Rx_3 \wedge, \ldots, \wedge x_mRx_1$.
2. (S, R) may have a root t. This means that for all $x \neq t$, there exists a (not necessarily unique) sequence $x_1, \ldots, x_m = x$ such that $tRx_1 \wedge x_1Rx_2 \wedge, \ldots, x_{m-1}R_x$.
3. Note that we have not imposed that S is finite, only finitary!
4. α is a function associating with each $x \in S$, a Dwff $\alpha(x)$ such that

(a) $\alpha(x)$ is atomic if x is an endpoint (i.e. $\neg\exists y(xRy)$).
(b) If x is not an endpoint then $\alpha(x) = \Downarrow \{\alpha(y)|xRy\}$.

5. Note that for the definition of (2) to be alright we need that (S, R, t) is acyclic and finitary. In fact the relation R needs to be well founded for α to exist!

Example (Canonical ordering of Dwffs).

1. Let $Q = \{q_1, \ldots, q_k\}$ be atoms. Let Θ_Q be the set of all Dwffs built up from $\{q_1, \ldots, q_k\}$. Define R on Θ_Q by
   - ARB iff B is an immediate subformula of A.[3]

   Then (Θ_Q, R) is acyclic and for each non-atomic A we have
   $$A = \Downarrow \{B|ARB\}.$$
2. For Q = set of all atoms, let $\Theta = \Theta_Q$. Then (Θ, R) is the canonical ordering of all Dwffs.

Definition 1.5 (*Models*). Let (S, R, α) be a decorated acyclic ordering.

1. A function h assigning a value $h(q) \in \{0, 1\}$ to any atom q of the language is called a model. In the case of three valued logic, h assigns values in $\{0, \frac{1}{2}, 1\}$.
2. Given a model h we can use it to give a truth value in $\{0, 1\}$ to every formula and node in (S, R, α) as follows (call this function $\mathbf{f}(x)$ for $x \in S$).
   (a) For x an endpoint let $\mathbf{f}(x) = h(\alpha(x))$.
   (b) for x not an endpoint let
   $$\mathbf{f}(x) = 1 - \max_{(xRy)}(\mathbf{f}(y))$$
3. Note that this definition follows exactly the truth table of $\Downarrow$. We have
   $$\alpha(x) = \Downarrow \{\alpha(y)|xRy\}.$$
   So assuming $\mathbf{f}(y)$ is the truth value of $\alpha(y)$ under h (in the model h), then $\mathbf{f}(x)$ would be the truth value of $\alpha(x)$ in the model h.

1.2. Semantic Tableaux for $\Downarrow$

We now give a tableaux formulation for our connective $\Downarrow$. We assume two valued logic. The tableaux would be slightly different for three valued logic. We first recall the usual tableaux formulation of classical logic with say, $\neg$ and $\wedge$.

---

[3] We have a notational compatibility problem. In argumentation one writes xRy to say that x attacks y. In logic we write ARB to say that B is an immediate subformula of A. The problem is that later we will have that the immediate subformula B of A actually attacks A.

To avoid confusion, let $\breve{R}$ be the converse relation of R So we have

$$xRy \text{ iff (definition) } y\breve{R}x$$

or using the ordered pair notation

$$(x, y) \in R \text{ iff } (y, x) \in \breve{R}$$

In the sequel we shall use either R or $\breve{R}$ depending on context. The reader should always remember that when we write $y\breve{R}x$ or xRy, then y attacks x.

In our figures we display xRy by $y \to x$.

The basic computational units are tables of the form

$$\tau = [A_1, \ldots, A_n \| B_1, \ldots, B_k]$$

The A_i are on the left and the B_j are on the right. We are seeking an assignment h to the atoms such that $h(A_i) = 1$ and $h(B_j) = 0$. So what is on the left is intended to be true and what is on the right is intended to be false.

In the middle of the computation we have a set $\mathbb{T}$ of such tableaux. We pick a tableau $\tau \in \mathbb{T}$ and replace it by new tableaux one or more, say $\tau_1, \ldots, \tau_m$ by performing certain tableaux operations. We get a new set of tableaux $\mathbb{T}'$. The new tableaux $\tau_1, \ldots, \tau_m$ are of less complexity than τ.

Depending on what we need the tableaux system for, we may or may not remember the connection between τ and $\tau_1, \ldots, \tau_m$.

The above description is not mathematically precise, it is only to remind the reader. Our definition of tableaux for $\Downarrow$ will be precise.

The following are the tableaux rules for $\neg$ and $\wedge$ of classical logic, just to remind the reader.

($\neg l$) Rule for $\neg$ on the left. Replace

$$\tau = [A_i, \neg A \| B_1, \ldots, B_k]$$

by

$$\tau' = [A_i \| B_1, \ldots, B_k, A]$$

($\neg r$) Rule for $\neg$ on the right. Replace

$$\tau = [A_1, \ldots, A_n \| B_1, \ldots, B_k, \neg B]$$

by

$$\tau' = [A_1, \ldots, A_n, B \| B_1, \ldots, B_k]$$

($\wedge l$) Rule for $\wedge$ on the left. Replace

$$\tau = [A_1, \ldots, A_n, C \wedge D \| B_1, \ldots, B_k]$$

by

$$\tau_1 = [A_1, \ldots, A_n, C, D \| B_1, \ldots, B_k]$$

($\wedge r$) Rule for $\wedge$ on the right. Replace

$$\tau = [A_1, \ldots, A_n \| B_1, \ldots, B_k, C \wedge D]$$

by the two tableaux

$$\tau_1 = [A_1, \ldots, A_k \| B_1, \ldots, B_k, C]$$

and

$$\tau_2 = [A_1, \ldots, A_n \| B_1, \ldots, B_k, D]$$

Closure of a tableau. A tableau

$$[A_1, \ldots, A_n \| B_1, \ldots, B_k]$$

is closed cf. for some i, j

$$A_i = B_j.$$

Definition 1.6 (*Tableaux for* $\Downarrow$).

1. A unit tableau for $\Downarrow$ has the form

$$\tau = [A_1, \ldots, A_n \| B_1, \ldots, B_k]$$

where A_i, B_j are Dwffs.

The tableau is *closed* if for some $i, j, A_i = B_j$.

2. Reduction rules for tableaux. Given a tableau τ, we perform possible applicable rules on τ to obtain new tableaux from it. At each step we apply exactly one rule. If the result of the application of the rule are tableaux $\tau_1, \ldots, \tau_n$, then we write $\tau\rho\tau_1, \ldots, \tau\rho\tau_n$.

We now just define the rules. The actual construction process is defined later. ($\Downarrow l$) **Rule for $\Downarrow$ on the left**
Replace

$$\tau = [A_1, \ldots, A_n, \Downarrow \{C_1, \ldots, C_m\} \| B_1, \ldots, B_k]$$

by

$$\tau' = [A_1, \ldots, A_n \| B_1, \ldots, B_k, C_1, \ldots, C_m]$$

write $\tau\rho\tau'$.

($\Downarrow r$) **Rule for $\Downarrow$ on the right**
Replace

$$\tau = [A_1, \ldots, A_n \| B_1, \ldots, B_k \Downarrow \{C_1, \ldots, C_m\}]$$

by $\tau_i, i = 1, \ldots, m$ where

$$\tau_i = [A_1, \ldots, A_n, C_i \| B_1, \ldots, B_k]$$

Write $\tau\rho\tau_i$ for $i = 1, \ldots, m$.

Note that if $\tau\rho\tau'$, the complexity of τ' is less than that of τ!

3. Given an initial family of tableaux

$$\mathbb{T}_0 = \{\tau, \tau', \tau'', \ldots\}$$

With relation $\rho_0 = \varnothing$, we build by induction a family of tableaux $(\mathbb{T}_n, \rho_n)$, with ρ_n a tree relation on $\mathbb{T}_n$, as follows.

Step $n + 1$
Assume that $(\mathbb{T}_n, \rho_n)$ has been defined. Assume that ρ_n is a tree relation such that the following holds:

(*) If $\tau \in \mathbb{T}$ is not an endpoint, then there exists a rule ($\Downarrow r$) (or ($\Downarrow l$)) and a formula $\Downarrow \{C_1, \ldots, C_m\}$ which is at the right of τ (resp. at the left of τ) such that the ρ_n immediate successors of τ are exactly the result of the application of the rule on the above formula.

We now define $(\mathbb{T}_{n+1}, \rho_{n+1})$ as follows:

3.1. Choose any endpoint tableau τ in $\mathbb{T}_n$. If all the formulas in τ are atomic then look for another endpoint tableau.

3.2. Assume τ has a Dwff $\Downarrow \{C_1, \ldots, C_m\}$ on the left. Let τ' be the result of applying the $(\Downarrow l)$ rule to this formula. Let $\mathbb{T}_{n+1} = \mathbb{T}_n \cup \{\tau'\}$ and let $\rho_{n+1} = \rho_n \cup \{(\tau, \tau')\}$.

3.3. Assume τ has a Dwff $\Downarrow \{C_1, \ldots, C_m\}$ on the right. Let $\tau'_1, \ldots, \tau'_m$ be the resulting tableaux from applying the $(\Downarrow r)$ rule to τ.
Let $\mathbb{T}_{n+1} = \mathbb{T}_n \cup \{\tau'_i\}$ and let ρ_{n+1} be $\rho_n \cup \{(\tau, \tau'_i) | i = 1, \ldots, m\}$.

3.4. Items (3.1)–(3.3) above are the steps which define $(\mathbb{T}_{n+1}, \rho_{n+1})$ from $(\mathbb{T}_n, \rho_n)$ through the use of an arbitrary choice of an endpoint in $(\mathbb{T}_n, \rho_n)$. Clearly it satisfies (*).

3.5. If the original $\mathbb{T}_0$ was infinite, then we want a fair sequence of choices, which will not neglect any endpoint tableaux.

4. Let $(\mathbb{T}_\infty, \rho_\infty) = \bigcup_n (\mathbb{T}_n, \rho_n)$. We get a tree. Let Π be a maximal path in the tree. Define two sets Π_r and Π_l as follows:

$$\Pi_l = \{A | \text{Dwff } A \text{ is the left side of some } \tau \text{ in } \Pi\}$$
$$\Pi_r = \{A | \text{Dwff } A \text{ is in the right hand side of some } \tau \text{ in } \Pi\}$$

Π is said to be admissible if $\Pi_r \cap \Pi_l = \varnothing$.

5. Let Π be admissible. Then the following holds:
($\sharp$) If $\Downarrow \{C_1, \ldots, C_m\} \in \Pi_l$ then $C_1, \ldots, C_m \in \Pi_r$.
Furthermore, if C_i is not atomic, i.e. $C_i = \Downarrow \{D^i_1, \ldots, D^i_{k(i)}\}$ then for some $j, D^i_j \in \Pi_l$.

1.3. Axiom System for $\Downarrow$

The logic of Dagger is not difficult to axiomatise. When we write axioms for a consequence relation of the form $\Delta \vdash A$, the elements of Δ are together as a conjunction. We also have negation because $\Downarrow A$ is $\neg A$. So basically we have what we need.

Definition 1.7 (*Axioms for* $\Downarrow$). Consider the following rules as a consequence relation for the language with $\Downarrow$. Δ is a finite set of Dwffs.

1. **Reflexivity**
$\Delta \vdash A$ if $A \in \Delta$
2. **Monotonicity**
$$\frac{\Delta \vdash A}{\Delta, B \vdash A}$$
3. **Cut**
$$\frac{\Delta, A \vdash B; \Delta, \Downarrow A \vdash B}{\Delta \vdash B}$$
4. $\Downarrow \Delta \vdash \Downarrow A$, if $A \in \Delta$
5. $\Downarrow A_i, i = 1, \ldots, n \vdash \Downarrow \{A_i \mid i = 1, \ldots, n\}$
6. $\Delta, A \vdash B$ iff $\Delta, \Downarrow B \vdash \Downarrow A$
7. $\Delta, \Downarrow\Downarrow A \vdash A$.

Definition 1.8. We can define $\Delta \vdash A$ for Δ infinite as well.
Let $\Delta \vdash A$ iff for some finite $\Delta_0 \subseteq \Delta$ we have $\Delta_0 \vdash A$.

Definition 1.9. A set Θ of wffs is consistent if for no Dwff A and finite $\Delta \subseteq \Theta$ do we have $\Delta \vdash A$ and $\Delta \vdash\Downarrow A$.

A theory is complete iff for any A, either $A \in \Delta$ or $\Downarrow A \in \Delta$.

Lemma 1.10. *Every finite consistent theory can be extended to a complete theory.*

Proof. The proof follows the usual lines known for classical logic. We first show that if Δ_0 is consistent and B a wff then either $\Delta_0 \cup \{B\}$ or $\Delta_0 \cup \{\Downarrow B\}$ are consistent. For otherwise for some A_1 and A_2 and finite $\Delta_0^1, \Delta_0^2 \subseteq \Delta_0$, we have

1. $\Delta_0^1, B \vdash A_1$
2. $\Delta_0^1, B \vdash\Downarrow A_1$
3. $\Delta_0^2, \Downarrow B \vdash A_2$
4. $\Delta_0^2, \Downarrow B \vdash\Downarrow A_2$.

From (1) and axiom (2) and axiom (6) we get for $\Delta_0' = \Delta_0^1 \cup \Delta_0^2$

$$\Delta_0', \Downarrow A_1 \vdash\Downarrow B$$
$$\Delta_0', A_1 \vdash\Downarrow B$$

And by cut rule we get

5. $\Delta_0' \vdash\Downarrow B$

From (3) and (4) and axiom (6), we get

$$\Delta_0', \Downarrow A_2 \vdash B$$
$$\Delta_0', A_2 \vdash B$$

And by cut we get

6. $\Delta_0' \vdash B$.

Thus Δ_0 is not consistent, a contradiction.

We can now follow the usual construction and extend any finite theory Δ_0 to a complete theory. Enumerate all wffs of the language $A_1, A_2, A_3, \ldots$.

Let

$$\Delta_{n+1} = \begin{cases} \Delta_n \cup \{A_n\} \text{ if consistent} \\ \Delta_n \cup \{\Downarrow A_n\}, \text{ otherwise} \end{cases}$$

By what we have just proved, Δ_{n+1} is consistent.

Let $\Delta_\infty = \bigcup_n \Delta_n$.

Δ_∞ is consistent for otherwise for some finite $\Delta' \subseteq \Delta_\infty$ and some X we have $\Delta' \vdash X$ and $\Delta \vdash\Downarrow X$.

Since Δ' is finite, then for some $n, \Delta' \subseteq \Delta_n$ contradicting the consistency of Δ_n. □

Theorem 1.11 (Completeness theorem). *The system of Definition 1.7 is sound and complete for the classical two valued semantics for* $\Downarrow$.

Proof. 1. Soundness can be seen from interpreting

$$\Downarrow \Delta = \bigwedge_{A \in \Delta} \neg A$$

2. To show completeness, let Δ_0 be a finite consistent theory. Extend Δ_0 to a complete theory Δ. Define a model h by

(*) $h(q) = 1$ iff $q \in \Delta, q$ atomic.

We now show by induction that for any $\Downarrow \{A_1, \ldots, A_n\}$ we have

(*) $h(\Downarrow \{A_1, \ldots, A_n\}) = 1$ iff $\Downarrow \{A_1, \ldots, A_n\})$ is in Δ.

Case 1. Assume $h(\Downarrow \{A_1, \ldots, A_n\}) = 1$. We show $\Downarrow \{A_1, \ldots, A_n\} \in \Delta$.

From the assumption we get that $h(A_i) = 0, i = 1, \ldots, n$. Hence by the induction hypothesis, $A_i \notin \Delta$ and hence $\Downarrow A_i \in \Delta, i = 1, \ldots, n$.

Therefore for some $\Delta_m, \Downarrow A_i \in \Delta_n, i = 1, \ldots, n$.

From axiom 5 we have $\Delta_n \vdash \Downarrow \{A_1, \ldots, A_n\}$. We claim $\Downarrow \{A_1, \ldots, A_n\} \in \Delta$. Otherwise $\Downarrow\Downarrow \{A_1, \ldots, A_n\} \in \Delta$ and so for some large m' $\Downarrow \{A_1, \ldots, A_n\} \in \Delta_{m'}$ and $\Downarrow\Downarrow \{A_1, \ldots, A_n\} \in \Delta_{m'}$, contradicting the consistency of $\Delta_{m'}$.

Case 2. Assume $\Downarrow \{A_1, \ldots, A_n\} \in \Delta$. Show $h(\Downarrow (A_1, \ldots, A_n)) = 1$. Otherwise, by the induction hypothesis for some $i, A_i \in \Delta$. Thus for some $m, A_i \in \Delta_m$. For some large enough $k \geq m$ we also have $\Downarrow \{A_1, \ldots, A_n\} \in \Delta_k$ and hence $\Downarrow A_i \in \Delta_k$ by axiom 4, contradicting the consistency of Δ_k.

Thus h is our required model because $h(\Delta_0) = 1$, since $\Delta_0 \subseteq \Delta$, and (*) holds.

This concludes the completeness theorem. □

2. Argumentation Frames

This section introduces the basic notions of argumentation frames and then proceeds to show equivalence with classical propositional logic.

We must make our methodology absolutely clear. We begin by comparing a very restricted form of argumentation frames, called nearly acyclic orderings (Definition 2.5), with the Dagger logic of Sect. 1. Such frames can be regarded as logic models or as argumentation models at the same time. Using that we will map some correspondences between logical concepts and argumentation concepts, in Sects. 2.2 and 2.3.

Once we have some idea on what may correspond to what, we start the technical machinery in earnest in Sect. 3 and generalise in Sects. 4 and 5.

2.1. Argumentation Preliminaries

An *argumentation framework* [18] consists of a set of arguments and an attack relation on these arguments. We only consider finite argumentation frameworks.

We need to make a notational comment. We used in Sect. 1 the notation:

Definition 2.1. An *argumentation framework* is a pair $(S, \breve{R})$ where S is a set and $\breve{R} \subseteq S \times S$. Let R be the converse relation of $\breve{R}$

We say that an argument A *attacks* an argument B iff $(A, B) \in \breve{R}$ or, equivalently, iff $(B, A) \in R$.

An argumentation framework can be represented as a directed graph in which the arguments are represented as nodes and the attack relation is

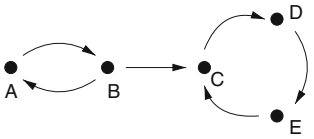


FIGURE 1. An argumentation framework represented as a directed graph (This figure originates with Martin Caminada, it is one of his favourites)

represented as arrows. For instance, the argumentation framework where $S = \{A, B, C, D, E\}$ and $R = \{(A, B), (B, A), (C, B), (C, D), (D, E), (E, C)\}$ is represented in Fig. 1.

The shorthand notation A^+ and A^- stands for, respectively, the set of arguments attacked by A and the set of arguments that attack A. Likewise, if $\mathcal{A}rgs$ is a set of arguments, then we write $\mathcal{A}rgs^+$ for the set of arguments that is attacked by at least one argument in $\mathcal{A}rgs$, and $\mathcal{A}rgs^-$ for the set of arguments that attack at least one argument in $\mathcal{A}rgs$. In the definition below, $F(\mathcal{A}rgs)$ stands for the set of arguments that are acceptable in the sense of [18].

Definition 2.2 (*Defense/conflict-free*). Let (S, R) be an argumentation framework, $A \in S$ and $\mathcal{A}rgs \subseteq S$. We define A^+ as $\{B \mid A \text{ attacks } B\}$ and $\mathcal{A}rgs^+$ as $\{B \mid A \text{ attacks } B \text{ for some } A \in \mathcal{A}rgs\}$. We define A^- as $\{B \mid B \text{ attacks } A\}$ and $\mathcal{A}rgs^-$ as $\{B \mid B \text{ attacks } A \text{ for some } A \in \mathcal{A}rgs\}$. $\mathcal{A}rgs$ is *conflict-free* iff $\mathcal{A}rgs \cap \mathcal{A}rgs^+ = \varnothing$. $\mathcal{A}rgs$ *defends* an argument A iff $A^- \subseteq \mathcal{A}rgs^+$. We define the function $F : 2^S \to 2^S$ as $F(\mathcal{A}rgs) = \{A \mid A \text{ is defended by } \mathcal{A}rgs\}$.

In the definition below, definitions of grounded, preferred and stable semantics are described in terms of complete semantics. These descriptions are not literally the same as those provided by Dung [18], but as was first stated in [11] and [5], these are in fact equivalent to Dung's original versions of grounded, preferred and stable semantics.

Definition 2.3 (*Acceptability semantics*). Let $(S, \breve{R})$ be an argumentation framework and let $\mathcal{A}rgs \subseteq S$ be a conflict-free set of arguments.

- $\mathcal{A}rgs$ is *admissible* iff $\mathcal{A}rgs \subseteq F(\mathcal{A}rgs)$.
- $\mathcal{A}rgs$ is a *complete* extension iff $\mathcal{A}rgs = F(\mathcal{A}rgs)$.
- $\mathcal{A}rgs$ is a *grounded* extension iff $\mathcal{A}rgs$ is the minimal (w.r.t. set-inclusion) complete extension.
- $\mathcal{A}rgs$ is a *preferred* extension iff $\mathcal{A}rgs$ is a maximal (w.r.t. set-inclusion) complete extension.
- $\mathcal{A}rgs$ is a *stable* extension iff $\mathcal{A}rgs$ is a complete extension that attacks every argument in $S \backslash \mathcal{A}rgs$.
- $\mathcal{A}rgs$ is a *semi-stable* extension iff $\mathcal{A}rgs$ is a complete extension where $\mathcal{A}rgs \cup \mathcal{A}rgs^+$ is maximal (w.r.t. set-inclusion).

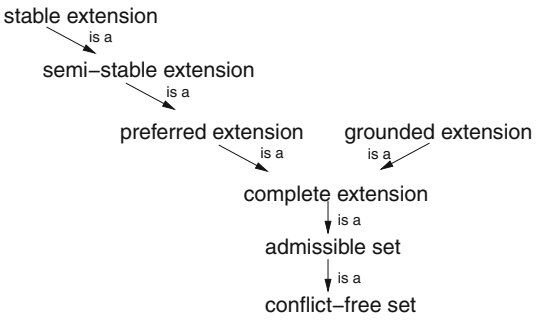


FIGURE 2. An overview of argumentation semantics (extension based), (this figure originates with Martin Caminada)

As an example, in the argumentation framework of Fig. 1 $\{B, D\}$ is a stable extension, $\{A\}$ is a preferred extension which is not stable or semi-stable, $\varnothing$ is the grounded extension, and $\{B\}$ is an admissible set which is not a complete extension.

It is known that for every argumentation framework, there exists at least one admissible set (the empty set), exactly one grounded extension, one or more complete extensions, one or more preferred extensions and zero or more stable extensions. Moreover, when the set of arguments in the argumentation framework is finite, there also exist one or more semi-stable extensions.

An overview of how the various extensions are related to each other is provided in Fig. 2. The fact that every stable extension is also a semi-stable extension, and that every semi-stable extension is also a preferred extension was first stated in [12]. All other relations shown in Fig. 2 have originally been stated in [18].

Definition 2.4 (*Caminada labelling*).

1. Let (S, R) be an argumentation frame. Let $\mathcal{A}rgs$ be a complete extension. Define a function $\lambda_{\mathcal{A}rgs}$ on S as follows:

   $\lambda_{\mathcal{A}rgs}(x) =$ in (or $=1$) if $x \in \mathcal{A}rgs$

   $\lambda_{\mathcal{A}rgs}(x) =$ out (or 0) if for some $y \in \mathcal{A}rgs, (x, y) \in R$

   $\lambda_{\mathcal{A}rgs}(x) =$ undecided (or $= \frac{1}{2}$) otherwise.

2. Note that it is clear that an extension is stable iff its λ function has codomain $\{0, 1\}$, i.e. it is a $\{0, 1\}$ valued function.

2.2. Discussion of Challenges

It is clear from the above that although both classical logic and argumentation frames use orderings as a basis, they deal with them in completely different ways. Let us see if we can get these two areas closer together. Consider the typical situation of Fig. 3.

Imagine that this is an ordering as part of the canonical ordering of Dwffs as in Example 1.1. We have $ARq_1 \wedge \ldots \wedge ARq_n$.

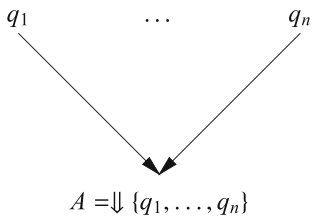


FIGURE 3.

We can also regard this figure as an argument frame, by taking the attack relation to be the converse relation $\breve{R}$ of R.

According to classical logic for $\Downarrow$ we have

- $\mathbf{f}(A) = 1$ iff $\bigwedge_{i=1}^{n} \mathbf{f}(q_i) = 0$

According to argumentation frames we have for any stable extension $\mathcal{A}rgs$:

- A is in iff $\bigwedge_{i=1}^{n}$ (q_i is out).
  or
- $\lambda_{\mathcal{A}rgs}(A) = 1$ iff $\bigwedge_{i=1}^{n}(\lambda_{\mathcal{A}rgs}(q_i) = 0)$.

The formula is the same. If we identify $\mathbf{f}(x) = 1$ as "x is in", then we can write with some abuse of notation that

- $\mathbf{f}(A) = 1 - \max(\mathbf{f}(q_i))$
- $\lambda_{\mathcal{A}rgs}(A) = 1 - \max(\lambda_{\mathcal{A}rgs}(q_i))$

Obviously we recognised that the recipe for being in any extension, given by Dung, corresponds to the truth table for $\Downarrow$.

There are, however, further differences. We now list them and then address and overcome them.

(D1) *From argumentation into logic.*
In classical logic the ordering should be acyclic. In argumentation we can have any binary relation, e.g. as in Fig. 1. We ask: how can we regard an arbitrary argumentation network as logic?

(D2) *From logic into argumentation.*
In logic the endpoints (i.e. $\neg\exists y(xRy)$) x can be assigned any value. In argumentation the endpoints can be value 1 (they are in).

If we consider Fig. 3, the endpoints $q_1, \ldots, q_n$ can be assigned by h (the assignment) any value, for example

$$h(q_1) = h(q_2) = \cdots = h(q_n) = 0.$$

In this case A will get value 1.

Argumentation theory, when viewing the ordering of Fig. 3, will allow only value 1.

$$h(q_1) = h(q_2) = \cdots = h(q_n) = 1.$$

(D3) In classical logic the values one gets are always 0 or 1. In argumentation we can get values 0,1 and undecided (in case of loops).

Let us address these points one by one.

(S1) Solution to D1
Consider the network in Fig. 1, how can we regard it as Logic? We now explain:

Almost in every elementary textbook in logic, there are the following exercises:

Exercise 1. The statement a says I am false (liar paradox). Can you give it a truth value?
The answer is no, we cannot. If we give it $\top$ then it should be $\bot$ and if we give it $\bot$ then it should be $\top$.

Exercise 2. You have two statements, a and b. a says b is false and b says a is false, can you give them a consistent truth value assignments?
The answer is that there are two possibilities;

$$a = \top \text{ and } b = \bot$$

and

$$a = \bot \text{ and } b = \top$$

Exercise 3. We have n statements, $a_1, \ldots, a_n$.

$$\begin{gathered} a_1 \text{ says that } a_2 \text{ is false} \\ \vdots \\ a_{n-1} \text{ says that } a_n \text{ is false} \\ a_n \text{ says } a_1 \text{ is false.} \end{gathered}$$

Can you consistently give these statements truth values?
The answer is that if n is odd we cannot but if n is even there are two solutions.

The above Exercises correspond to argumentation networks
Exercise 1 corresponds to the network containing a single node a attacking itself.
Exercise 2 corresponds to the network with two nodes a and b, where a attacks b and b attacks a.
Exercise 3 corresponds to the network which is a cycle of n nodes, as described below, where $\rightarrow$ is attack:

$$a_1 \rightarrow a_2 \rightarrow \cdots \rightarrow a_n \rightarrow a_1.$$

The logic exercise question:

what consistent truth value assignments can you give to the statements, including no assignment to some?

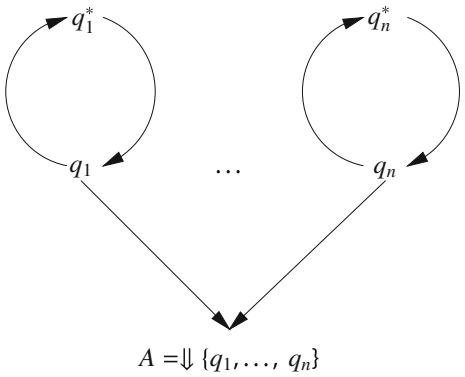


FIGURE 4.

Corresponds to the question:

what are the extensions to the network, including undecided?

Now let us look at the network of Fig. 1 and turn it into a logic exercise.

Exercise 4. Given the following statements, can you give them consistent truth value assignments (including no assignment to some)?

A says B is false

B says A is false and C is false

C says D is false

D says E is false

E says C is false.

I think by now the reader has a fair idea of how we regard an argumentation network as an exercise in logic.

By the way, some logic textbooks have some hard exercises corresponding to some pretty complicated networks!

For general formal construction, see the subsection on Boolean Networks below, especially Theorem 2.11 and Remark 2.12

(S2) Solution to D2

We still have the problem of the value assignments to endpoints. Argumentation theory requires us to give them value 1. This can be overcome by splitting every endpoint x into 2 points x and x^*, with x^*Rx and xRx^*.

So Fig. 1 becomes Fig. 4.

This figure is not meaningful from the logic point of view. But from the argumentation point of view we can have 2^n stable extensions and 3^n complete extensions. For the case of two valued logic we use only stable extensions. Let E be any stable extension. Let $h_E(q_i) = 1$ iff $q_i \in E$. Now we have 2^n assignments h to the atoms.

For the case of three valued logic, we use complete extensions. Let E be a complete extension, then let $h_E(q_i) = 1$ if q_i is in E, let $h_E(q_i) = 0$ if q_i^* is in E and let $h_E(q_i) = \frac{1}{2}$ if neither is in E.

For the moment, this discussion is only intuitive. We need to set up the formal machinery to explain why and how we can pair Figs. 3 with 4.

The perceptive reader might ask: in what way does the discussion in (S2) address the difficulty (D2)?
Is it not just a trick?
We are saying:
"the graph of Fig. 3 doesn't work, so let's switch to the supergraph of Fig. 4".
The answer is that no, it is not a trick; it is common practice in mathematics to translate one system into another. See for example http://en.wikipedia.org/wiki/Complex_number#Matrix_representation_of_complex_numbers, for the translation of complex numbers into matrices.

In our case we are translating Fig. 3, which is a figure in the area of logic, into Fig. 4, which is a figure in the area of argumentation networks.

Take the letter q_1 in Fig. 3; here we are in the realm of logic, and q_1 is an atom, so we can talk about substituting another logical formula B for q_1.

Take now q_1 in Fig. 4; here we are in the realm of argumentation networks, q_1 is an abstract argument and the question of substituting a formula B of logic for q_1 is meaningless.

By the way, we have investigated in [22] the possibility of substituting another argumentation network for q_1, but this is new research!

To make this point crystal clear, recall our answer (S1) above, as to how we translate from argumentation networks into logic. This is the other direction of showing that argumentation networks are equivalent to logic.

For a full discussion see Sect. 6.

(S3) Solution to D3
To solve the problem of undecided values in argumentation we can proceed in two ways

(a) Restrict argumentation extensions to stable extensions, or more specifically allow for orderings that give rise to stable extensions. Fortunately the orderings needed to solve (D2) are such orderings (e.g. Fig. 4). This is not, however, really a solution to (D3) because we are making the problem disappear. Much better is:
(b) Allow and move to a 3-valued logic for the connective $\Downarrow$. This option is already incorporated into our definitions above.

We shall follow both possibilities. We shall address (b) in Sect. 3.

Definition 2.5 (*Nearly acyclic ordering*). An ordering (S^*, R^*) is said to be nearly acyclic iff it is obtained from an acyclic ordering (S, R) as follows.

1. Let T be all the endpoints of (S, R), i.e. $T = \{x | \neg \exists y \in S x R y\}$.
2. Let T^* be a set of new points of the form

$$T^* = \{x^* | x \in T\}.$$

Let

$$S^* = S \cup T^*$$

and

$$R^* = R \cup \{(x, x^*), (x^*, x)\}.$$

3. We say that (S, R) and (S^*, R^*) are paired.

2.3. Integrating Argumentation Frames with Classical Logic: An Informal View

We are now ready to present in principle (technical definition will be given later) our method of integration. The idea is very simple. Present two networks (S, R) and (S^*, R^*) with $S \subseteq S^*, R \subseteq R^*$ such that they form a pair as in Definition 2.5. We can then view (S, R) either as classical logic formula construction tree as done in Sect. 1.1 or as an argumentation frame, a subframe of (S^*, R^*), and do what is natural to do from each point of view and compare what each movement means from the other point of view.

Definition 2.6 (*The canonical argumentation frame based on Dagger logic*). Our starting point is the general canonical model (Θ, R) of Example 1.1.

The elements of Θ are all Dwffs of the language of classical logic based on $\Downarrow$ and the atoms Q. R is defined by

- ARB iff B is an immediate subformula of A.

Recall that this means that

- $\Downarrow (A_1, \ldots, A_n) R A_i, i = 1, \ldots, n$.

Regarding (Θ, R) as just an acyclic ordering, we can construct its pair $\Theta^* \supseteq \Theta, R^* \supseteq R$ as in Definition 2.5.

We add a set

$$Q^* = \{q^* | q \in Q\}$$

that is $\Theta^* = Q^* \cup \Theta$, and $R^* = R \cup \{(q, q^*), (q^*, q)\}$.

We now compare (Θ, R) from two points of view.

1. From the point of view of logic, when viewed as a canonical model for all Dwffs of the language of $\Downarrow$ and where we allow for arbitrary assignments h to the atoms of Q.
2. From the point of view of argumentation theory, where $(\Theta^*, \breve{R}^*)$ is regarded as just an ordering, just an argumentation frame in which $\breve{R}$ is the converse relation of R and where we examine the effects of argumentation extensions of $(\Theta^*, \breve{R}^*)$ on the subnetwork $(\Theta, \breve{R})$.

We now go through the concepts one by one.

(C1) **Concept of attack**

B attacks A in the argumentation network means that B is an immediate subformula of A in logic. Note that since

$$A = \Downarrow (B_1, \ldots, B_n) = \bigwedge_{i=1}^{n} \neg B_i$$

when B_i attacks A we have that $\vdash \neg(A \wedge B_i)$.

Thus this concept corresponds to the inconsistency attacks used by Besnard and Hunter in their book [6]. Note for example that $\neg\neg B_i = \Downarrow\Downarrow B_i$ does not attack A because it is not an immediate subformula of A.[4]

(C2) **The concept of an extension**

A complete extension in (Θ^*, R^*) can be generated by any function $\lambda_0 : Q \to \{0, \frac{1}{2}, 1\}$ generating a subset of Q of all q in Q which get value 1, a subset of Q^* of all q^* such that the value of q is 0 and the rest get value $\frac{1}{2}$.

The ordering (Θ^*, R^*) allows for stable extensions. Stable extensions give rise to $\{0, 1\}$ values on Q and Q^*. So let E^* be any stable extension. E^* must choose one from any mutually attacking pair q, q^*. So let $\lambda_0(q) = 1$ iff $q \in E^*$.

The values $\lambda_*(A), A \in \Theta$ are now determined uniquely by λ_0.

We have

$$\lambda_*(\Downarrow (A_1, \ldots, A_n\}) = 1 - \max_i \lambda_*(A_i).$$

Now let us look at λ_0 as a logical assignment $h = \lambda_0$ to the atoms.

For the two valued case we have, after propagating truth values, that (see Definition 1.5)

- $\Downarrow \{A_1, \ldots, A_n\}$ holds at the model $h = \lambda_0$ iff for all i, A_i does not hold in h.

The above means that we have the correspondence

- stable extension in argumentation corresponds to a complete theory (or equivalently a model) in classical and two valued logic.

For the three valued case it is slightly more complicated but we get a similar situation. We need to define the notion of what is a complete three valued theory. This can be defined (we will do this in Sect. 3) and we have:

- complete extensions correspond to complete three valued theories in three valued logic

There are other types of extensions in argumentation as shown in Fig. 2, for orderings allowing for undecided elements. We shall deal with this in Sect. 3.

(C3) **The concept of a logical Dwff**

This corresponds to the notion of an argument node.

The notion of a logical theory therefore corresponds to the notion of a set of argument nodes.

[4] We believe we can draw conclusions from our paper to the debate about the inferential argumentation frames, as championed by Martin Caminada. We need more time to assess our results. See, however, Sect. 3.3.

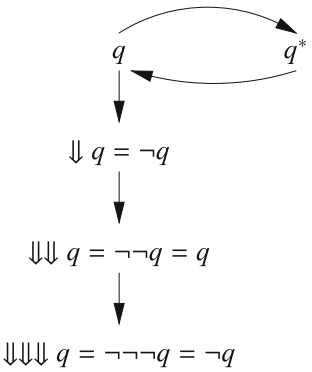


FIGURE 5.

(C4) **The concept of consequence $A \vdash B$ in logic**
This means that in any model of A is a model of B.

Since what corresponds to "model" is "stable extension" for two valued logic and a "complete extension" for three valued logic, the corresponding notion is as follows:

- Let (S, R) be an argumentation frame. Let $x, y \in S$. We say $x \vdash_{(S,R)} y$ iff in any stable (resp. complete) extension E, if $x \in E$ then $y \in E$.

Similarly we have a correspondence between theories (and subsets of arguments) proving another Dwff (another argument).

(C5) **What corresponds in logic to conflict free set?**
This concept has no logical counterpart, as a conflict free set of formulas may be inconsistent, as Fig. 5 shows.

The nodes q and $\Downarrow\Downarrow\Downarrow q$ are conflict free but inconsistent in logic.

The reader should not despair, this is to be expected. If we look at the literature on how Dung's abstract argumentation framework is actually instantiated (by various scholars) we see that what is needed to produce consistency is not conflict-freeness but the stronger condition of admissibility. The really important concept is that of an admissible set. See Caminada and Wu [13].

(C6) **The concept of admissible set**
This corresponds to the notion of an admissible path in the tableaux system for $\Downarrow$, as described in items (4) and (5) of Definition 1.6.

Let me do this systematically. Let E be an admissible set. It is

(1) Conflict free,
and
(2) If A attacks $B \in E$ then for some $C \in E, C$ attacks A.

Let us see what (1) and (2) mean in terms of logic. (Note that condition (1) alone does not have a logical meaning, as we saw in (C5) above, but (1) and (2) together do have a logical meaning)

(1) means that we do not have any $\Downarrow \{C_1, \ldots, C_n\}$ and any C_i both in E.

(2) means that if $B =\Downarrow \{A_1, \ldots, A_n\} \in E$ then for some $X_i \in E$, we have that X_i attacks A_i, for each $i = 1, \ldots, n$. This must hold since A_i attacks B.

We remember that $E \subseteq \Theta^*$. So if A_i is an atom q_i, this means $q_i^* \in E$. If $A_i =\Downarrow \{D_1^i, \ldots, D_{k(i)}^i\}$ then X_i is some $D_j^i \in E$.

In terms of tableaux think of the following set

$$E_l = \{X | X \in E \cap \Theta\}$$
$$E_r = \{X | \text{for some } Y \in E | XR^*Y, \text{ i.e } Y \text{ attacks } X\}.$$

We claim there is a path Π in the sense of Definition 1.6. We need to find the initial tableaux to start the process. We can take $[E_l \| E_r]$ as our tableaux and observe that the rules (1) and (2) are really tableaux rules which ensure a non-closed path. The conflict free property ensures that for each atomic q we cannot have both $q, q^* \in E$. (Remember E is admissible in (Θ^*, R^*)!)

So an admissible set corresponds to an admissible path in the tableaux proof restricted to the members of the set. There is a nicer way to do this by taking minimal elements of E but we do this in the next section.

We shall also address and discuss what happens in the case of the empty set being an admissible set. Is it the empty tableau?

(C7) **The logical concept of proof**

We now examine how to represent in an argumentation frame the notion of logical proof. This is a dynamic concept and obviously we shall have to do something dynamic; like execute a logical walk over the nodes of the frame.

According to (C3), a Dwff corresponds to a node in the argumentation frame. According to (C4) the notion of

$$\text{consequence } A \vdash B,$$

corresponds to the notion of

B is an element of any extension (of the correct type) which contains A

Thus if we have a step-by-step proof of B from A, i.e.

$$D_1 = A$$
$$\vdots$$
$$D_n = B$$

Then $(D_1, \ldots, D_n)$ is a logic "walk" along nodes. We must define the rules of such a "walk" in terms of the geometry of the argumentation frame, and show that such a "walk" exists iff $A \vdash B$.

Our starting point is modus ponens. This has the form

$$A, A \to B \vdash B$$

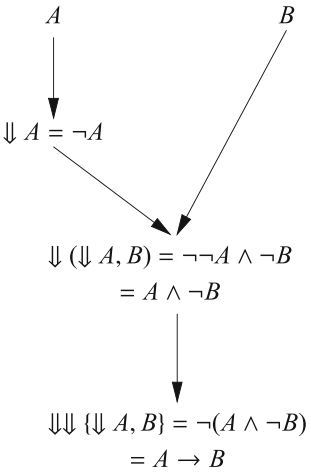


FIGURE 6.

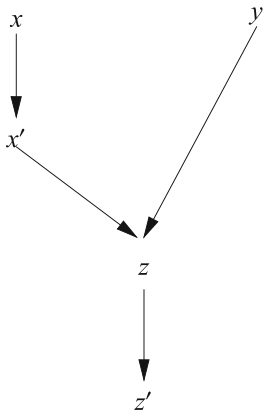


FIGURE 7.

The formula for $A \to B$ in the language of $\Downarrow$ is

$$\begin{aligned} A \to B &\equiv \neg(\neg\neg A \wedge \neg B) \\ &\equiv \Downarrow\Downarrow \{\Downarrow A, B\} \end{aligned}$$

Figure 6 shows what is happening

According to (C3), a logical theory is a set of argument nodes, i.e. a subset $T \subseteq S$. So if $A, A \to B \in T$, this means that we can expand T and let also $B \in T$.

We can therefore offer the following definition for geometrical modus ponens (GMP):

Definition 2.7 (*Geometrical modus ponens*).

(a) Let (S, R) be an argumentation frame. The pattern subframe of Fig. 7 is called modus ponens pattern. This pattern is directional. We can recognise the bottom z' and the left top x (distance 4 from the bottom) and the right top y, being the top at distance 3 from the bottom.

(b) The modus ponens closure rule on a set $T \subseteq S$ is according to this pattern the following:
- if $x, z' \in T$ then let $y \in T$.

(c) We now look at the $\rightarrow$ elimination rule.
To show $A \rightarrow B$, assume A and prove B
So geometrically to add the node z' of Fig. 7 to T we need to identify z' as part of the pattern of Fig. 7 and temporarily add the node x to T, show that the geometry allows us to add y to $T \cup \{x\}$. Then we discharge x and end up with $T \cup \{z'\}$.

(d) The concept of $T \mapsto x$ can be defined that starting from T we can follow a sequence of geometrical moves which allow us to expand T to T' containing x. An example will help.

Example. Consider the following derivation

(a) $B \rightarrow (A \rightarrow C)$, assumption
(b) A, assumption
(c) We aim to prove $B \rightarrow C$
(d) Show $B \rightarrow C$ from subproof
  (i) B, assumption
  (ii) $A \rightarrow C$, from (a) and (i)
  (iii) C from (b) and (ii)
  (iv) Exit subproof, discharge B.

Consider Fig. 8.

Look only at patterns. The wffs written is only to help read the pattern. We start with

$$T = \{y, e'\}$$

We want to show that $T \mapsto c$.

Step 0. Recognise that c is the bottom of a pattern $\{x, x', b, c, z\}$, with x left top and z right top.

Step 1. Add x as an assumption to T to form $T_1 = \{x, y, e'\}$.

Step 2. Using the GMP pattern $\{x, x', d, e, e'\}$ and the fact that $x, e' \in T_1$, we add d as an inferred item, to T_1 to form $T_2 = \{x, e', d, y\}$ (note that d stands for $A \rightarrow C$).

Step 3. Using the pattern $\{y, y', a, d, z\}$ we can add z to T_2 to obtain $T_3 = T_2 \cup \{z\}$.

Step 4. Having started with $T \cup \{x\}$ and ended up with $T \cup \{x, z\}$, we can now add c to T because c is the bottom node of a modus ponens pattern whose top left is x and top right is z and we have shown that $T \cup \{x\} \mapsto z$.

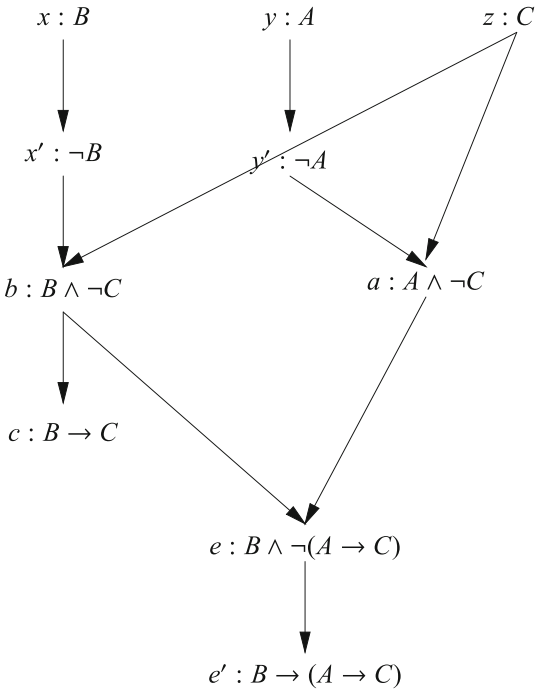


FIGURE 8.

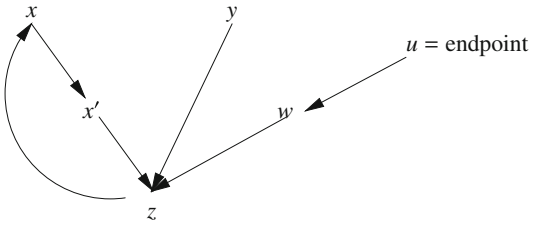


FIGURE 9.

Thus the proof is a sequence of nodes each obtained from T and previous members of the sequence using geometrical (GMP rules) considerations.

Remark 2.8. The reader should note that our discussion here is only an intuitive introduction, to get the idea of what a proof is. The exact definition of a GMP pattern need yet to be given in the correct context. To get an idea of what we mean, consider Fig. 9.

This figure may not conform to the pattern of Fig. 7. First we have x itself acting as z'. We may ignore that. After all if $A \equiv (A \to B)$ and we have A we can still get B. A more interesting problem is the presence of node w attacking z. This really breaks the pattern. However, w

is attacked by u which is an endpoint. So we do have the pattern if we take that into account.

We need a proper technical definition for our intuitions.

Remark 2.9 (Inference based argumentation and Geometrical Modus Ponens). We note that we can make a contribution to the debate about inference based argumentation. There is the view, championed by senior figures in the field among them Martin Caminada, Henry Prakken and others see [14] and [34], that there are three stages in constructing an argumentation network

1. start with base logic and a knowledge base, which can be used to construct proofs
2. Use the proofs from 1. to construct arguments, the attack relation and an argumentation frame
3. Use the network of 2. to construct extensions and expect the winning arguments in any extensions in 2. to give rise to a consistent theory of the base logic in 1.

See for example [13,34]. The notion of geometrical Modus Ponens allows us to offer an abstraction of step 1 above in the sense that we can include it in the abstract within step 2. See Sect. 3.3 below.

The advantages of doing this are two fold

- We continue Dung's spirit of abstraction and not regress backwards into specialised systems. If in step 1 we already have the detailed notions of base logic, proof and inconsistency, why add step 2 to dress it up in some high level concepts which are obvious and available already in step 1?
- The geometrical concepts of walking along a network to simulate the proof procedures are not just a local technical device cooked up especially to enable our local purpose—it is a commonly used apparatus of traversing graphs and networks used in many areas, including graph theory and Kripke semantics yielding a rich variety of results.

Summary of correspondence between logic and arguments

It would be helpful to summarise the correspondence between abstract argumentation and Dagger classical logic. See Table 2.

2.4. Boolean Networks

The main result of this section is to show that every argumentation network can be obtained in a systematic way from logic. In Sect. 2.2, under (S1), we discussed this. We intuitively shown in (S1) that every argumentation network can be obtained from a logic exercise. This however, is not mathematically systematic.

We show a stronger result, that every Boolean network can be obtained from logic.

To motivate Boolean networks, let us start with an example of Brewka and Woltran.

The correspondence between classical Dagger logic and argumentation makes the results of Gabbay [22,24] and Brewka and Woltran [9,10], about

Table 2.

	Logic concept	Argumentation concept
1.	Classical logic formulated with $\Downarrow$	A specific canonical argumentation frame presented in Definition 2.6
2.	A is an immediate subformula of B	A attacks B
3.	Complete logical theory (or equivalently) a model	Complete extension for the case of 3-valued model, and a stable extension for the case of a two valued model
4.	A Dwff	An argument (a node in the argumentation frame)
5.	Consequence $A \vdash B$	Node b is present in any complete extension containing node a
6.	No direct correspondence	Conflict-free set
7.	A maximal path in the tableaux (based on the elements of the admissible extension)	Admissible extension
8.	Modus ponens or other proof rules	Geometrical patterns on the abstract argumentation network
9.	A proof sequence from A to B using certain proof rules	A "walk" in the network from point a to point b respecting and using certain geometrical patterns

translating an arbitrary Boolean network into argumentation network, rather predictable.

We simply translate the relevant Boolean formulas into the language with Dagger and they will automatically be embedded into our canonical argumentation frame (Θ^*, R^*) of Definition 2.6.

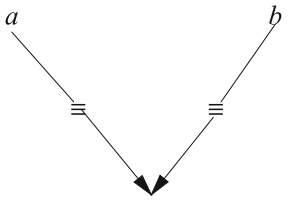
Let us begin with an example of Brewka and Woltran.

Example (Brewka and Woltran Boolean example). Brewka and Woltran [9] put forward the example in Fig. 10, in which nodes a and b are neither attacking nor supporting node c. In their abstract dialectical framework one can write Boolean conditions on the nodes. In this case we want $c \equiv [(a \wedge \neg b) \vee (\neg a \wedge b)]$, i.e. we want $c =$ in when *exactly one* of $\{a, b\}$ is in.

This example was addressed specifically in our paper [7, Example14]. It was translated into Dung argumentation networks using additional nodes. Paper [7] then follows to indicate using [22] how any abstract dialectical framework can be so translated into a Dung network.

We quote Figure 27 from [7] as our Fig. 11 and we implement it in Fig. 12.

If argument a is acceptable and argument b is acceptable then arguments u_a, u_b are not acceptable and argument y is acceptable and argument c is not acceptable.



$c = [(a \wedge \neg b) \vee (\neg a \wedge b)]$

We use —≡⟶ to indicate Boolean connections.

FIGURE 10.

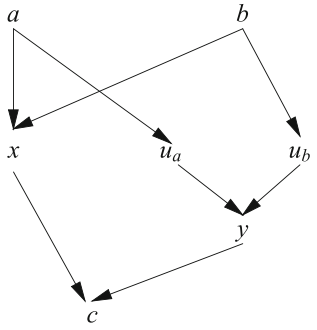


FIGURE 11.

If both arguments a and b are not acceptable then argument x is acceptable and so argument c is not acceptable.

If argument a is not acceptable and argument b is acceptable then argument x is not acceptable, argument u_a is acceptable and u_b is not acceptable. Because u_a is acceptable then we get y not acceptable. Since both x and x are not acceptable then we get argument c as acceptable, as desired.

If argument a is acceptable and argument b is not acceptable then we get argument x is not acceptable and argument u_b is acceptable and argument y is not acceptable. Thus argument c is acceptable, as desired.

The general case requires the equational algebraic approach, we address it in our paper [24].

Consider Fig. 13.

This presents a problem at the moment because it is cyclic.

Definition 2.10 (*Boolean networks*). A Boolean network has the form $(S, \breve{R}, \Psi_t)$, $t \in S$, where $\breve{R} \subseteq S \times S$ and for each t, Ψ_t is a Boolean wff in the set of atoms $\{y\breve{R}t\}$. For t an endpoint, Ψ_t is either $\top$ or $\bot$. We write $\Psi(y_1, \ldots, y_k)$, where $\{y_1, \ldots, y_k\} = \{y\breve{R}t\}$.

A solution to the Boolean network is a function h giving values in $\{\top, \bot\}$ to each node $t \in S$, such that the following holds

(*) $h(t) = \Psi_t(h(y_1), \ldots, h(y_k))$

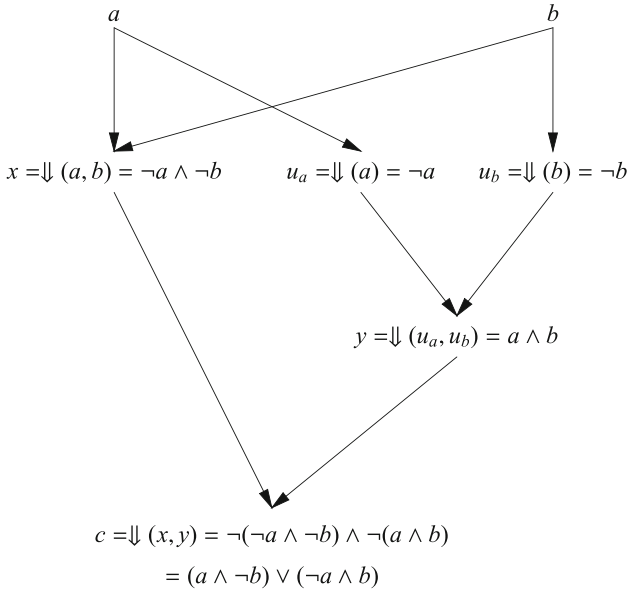


FIGURE 12.

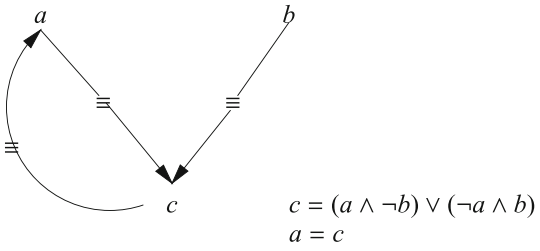


FIGURE 13.

Note that we may not have a solution. Take for example the network

$$(\{t\}, t\breve{R}t, \Psi_t = \neg t).$$

A solution for this network requires a function h such that

$$h(t) = \neg h(t).$$

This is not possible.

Theorem 2.11 (Representation theorem for Boolean networks). *Every finite acyclic Boolean network can be faithfully embedded into an argumentation network.*

Proof. Let $(S, \breve{R}, \Psi_t)$ be an acyclic network. Write each $\Psi_t(y_1, \ldots, y_k)$ as a Dwff in the y's using the language with $\Downarrow$. Since we are now using logic, we use the relation R instead of $\breve{R}$. Remember that one is the converse of the other. So y attacks x can be written either as xRy or as $y\breve{R}x$ or $y \to x$.

So our network has the form (S, R, Ψ_t^*) where Ψ_t^* is written with $\Downarrow$. (We will have to accept $\top$ and $\bot$ as formulas as well.)

Now define new formulas $\varphi_t^*, t \in S$ as follows:

1. For t an endpoint (i.e. $\neg\exists y(tRy)$) let $\varphi_t^* = \Psi_t^*$.
2. Let $t, y_1, \ldots, y_k$ be such that $\{y|tRy\} = \{y_1, \ldots, y_k\}$.

Let Ψ_t^* be written as $\Psi_t^*(y_1, \ldots, y_k)$. Then let

$$\varphi_t^* = \Psi_t^*(y_1/\varphi_{y_1}^*, \ldots, y_k/\varphi_{y_k}^*)$$

It is easy to see that

1. tRs iff φ_s^* is a subformula of φ_t^*.
2. For any solution h to (*) we have $h(t) = h(\varphi_t^*)$.

From the above it is clear that the subset $\{\varphi_t^*|t \in S\}$ is a subnetwork of the canonical network (Θ^*, R^*) based on the atoms S of Definition 2.6.

This proves the theorem. □

Remark 2.12 (Boolean network with loops). What do we do in the case of (S, R, Ψ_t) with loops where a solution h may or may not exist?

Again we assume Ψ_t is written with $\Downarrow$ only.

Define Dwffs $\Psi_t^n, n = 1, 2, \ldots$.

1. $\Psi_t^1 = \Psi_t$
2. $\Psi_t^{n+1} = \Psi_t(\{y/\Psi_y^n|tRy\})$.

We are getting an infinite number of wffs but all of these appear in the canonical model based on S as defined in Definition 2.6.

Note that for any solution h satisfying (*) we also have

$$h(\Psi_t^n) = h(\Psi_t).$$

Now consider the equivalence relation on the canonical model (Θ^*, R^*) as follows:

$x \approx_t y$ iff for some n, m we have that $x = \Psi_t^n \wedge y = \Psi_t^m$. Let us denote by $E(t)$ the set of all x such that $x \approx_t t$.

Let $x \approx y$ be defined by induction as follows

1. $x \approx y$ iff $x \approx_t y$ for some t
2. $\Downarrow \{x_i\} \approx \Downarrow \{y_i\}$, iff $x_i \approx y_i$, for $i = 1, 2, \ldots$.

$\approx$ is well defined. Take the factor $(\Theta^*/\approx, R^*/\approx)$ network with Θ^* being the equivalence classes and the attack relation $R^*/\approx$ is defined by

$$x/\approx R^*/\approx y/\approx \text{ iff for some } x' \approx x, y' \approx y \text{ we have } xR^*y.$$

We claim (S, R, Ψ_t) is faithfully embedded in $(\Theta^*/\approx, R^*/\approx)$.

Let t be mapped onto the equivalence class $\{\Psi_t^n\}$.

Assume tRs holds. Then $\Psi_t^{n+1} = \Psi_t(, \ldots \Psi_s^n \ldots)$ by definition. So for any h satisfying (*) we also have

$$h(\Psi_t/\approx) = \Psi_t(\{h(\Psi_s/\approx)|tRs\}.$$

Conversely if there is an h on the canonical model giving all elements of the same class the same value then there is an h satisfying (*) on the original Boolean network.

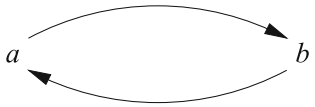


FIGURE 14.

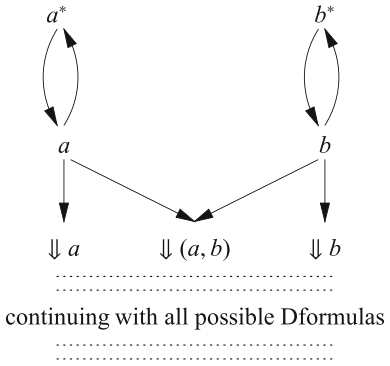


FIGURE 15.

Let us give some examples illustrating the process of Remark 2.12.

Example. Consider the loop in Fig. 14.

In this figure we have $\Psi_a = \neg b$ and $\Psi_b = \neg a$. We therefore have

$$\begin{array}{ll} \Psi_a^1 = \neg b & \Psi_b^1 = \neg a \\ \Psi_a^2 = a & \Psi_b^2 = b \\ \Psi_a^3 = \neg b & \Psi_b^3 = \neg a \\ \vdots & \vdots \end{array}$$

The initial equivalence classes are therefore

$$E(a) = \{a, \neg b\} \text{ and } E(b) = \{\neg a, b\}.$$

Consider the canonical model, based on the atoms $\{a, a^*, b, b^*\}$. Figure 15 describes part of it. Recall that $\neg A$ is $\Downarrow A$ in the Dagger language.

Taking into account the equivalence classes we get Fig. 16.

Example. Let us try an example where there is no solution in $\{0, 1\}$, as in Fig. 17.

We get

$$\begin{array}{lll} \Psi_a^1 = \neg c & \Psi_b^1 = \neg a & \Psi_c^1 = \neg b \\ \Psi_a^2 = b & \Psi_b^2 = c & \Psi_c^2 = a \\ \Psi_a^3 = \neg a & \Psi_b^3 = \neg b & \Psi_c^3 = \neg c \\ \Psi_a^4 = c & \Psi_b^4 = a & \Psi_c^4 = b \\ \Psi_a^5 = \neg b & \Psi_b^5 = \neg c & \Psi_c^5 = \neg a \end{array}$$

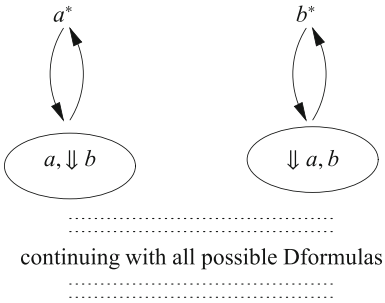


FIGURE 16.

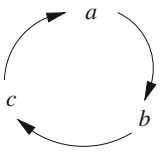


FIGURE 17.

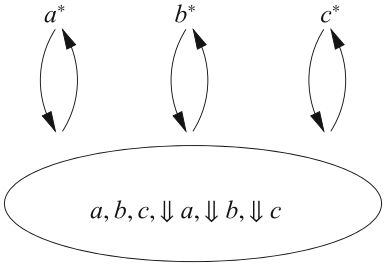


FIGURE 18.

$$\begin{array}{lll} \Psi_a^6 = a & \Psi_b^6 = b & \Psi_c^6 = c \\ \Psi_a^7 = \neg c & \Psi_b^7 = \neg a & \Psi_c^7 = \neg b \\ \vdots & \vdots & \vdots \end{array}$$

Obviously there is one equivalence class

$$\{a, b, c, \neg a, \neg b, \neg c\}$$

The graph we get is in Fig. 18.

There is no solution in $\{0, 1\}$. The solution in $\{0, 1, \frac{1}{2}\}$ is $a = b = c = a^* = b^* = c^* = \frac{1}{2}$, so also $\Downarrow a = \Downarrow b = \Downarrow c = \frac{1}{2}$.

3. Technical Results on Integrating Argumentation as Logic

3.1. What is a Logical System

Our first step is to clarify what we mean by logic. We give precise definitions of a view of "what is a logical system" which is most friendly to argumentation.

We now explain our ideas. Consider Fig. 11. This is just a network. How can we turn it into a logic? One obvious way of doing it is to go through a semantical interpretation. This is an ordering, a set of worlds with a binary relation and so we can regard it as a Kripke model for some logic. It can thus define a modal logic, or an intuitionistic intermediate logic, or a provability logic, etc., etc.

In our paper [21], we took this approach and interpreted argumentation networks into the modal logic of provability. We got surprising results about what is the logical content of argumentation network, but this does not present the network as a logic.

The construction in Sect. 1 of this paper does turn the network into a logic. It says the nodes t are formulas and the attack of $x_1, \ldots, x_n$ on t means "being an immediate subformula of". So Fig. 11 actually represents Fig. 12.

So far so good, the problem is that argumentation networks can have loops. So how do we interpret Fig. 9?

Our idea is to abstract from the actual syntax of a formula and regard it as an object and have the relations of "being an immediate subformula of" as the interpretation of the arrow. If we do that, all we have left is just an ordering? Where is the logic in it?

The answer is simple. All we need to add is the truth table functions which tell us for each node t and its attacking nodes $x_1, \ldots, x_n$, how to get the truth value of the formula t (whose structure we do not know) from the values of its immediate subformula $x_1, \ldots, x_n$, (whose structure we also do not know). Let us call such a truth propagating function

$$\mathbf{h}_t(x_1, \ldots, x_n).$$

We may not know the syntactic structure of the formulas but we do know how to calculate their values! When we know how to calculate values we do not mind cycles, they just give rise to equations to be solved.

Clearly $\mathbf{h}_t$ is also part of the logic.[5]

Now that we have the idea, all that remains is to figure out the coherent technical details. This we now do.

---

[5] The perceptive reader might be a bit puzzled here. He may reason as follows:

> If I know the immediate subformulas of t and the function yielding the truth value of t from the truth value of its immediate subformulas, then this means that the logic is truth-functional (at least there) and therefore this function characterizes the main connective of t, so I know the structure of t after all (and, recursively, I would know the structure of the subformulas, applying the same principle).

Note, however, that h_t can vary with t and may not be Boolean. We may also have loops in the network.

Definition 3.1 (*Function spaces*). By a function space we mean the following

1. A set V with a family of functions $\mathbb{F}$ of the form

$$\mathbf{h} : V^n \mapsto V$$

For $n = 1, 2, \ldots$.

The set V may have some operations on it by which means the functions $\mathbf{h}$ in $\mathbb{F}$ are defined. We give two examples:

(a) $V = [0, 1]$, the set of all real numbers between 0 and 1 and $\mathbb{F}$ is the set of all continuous functions with any number of variables.

(b) V is a complete partial order (i.e. it has a partial ordering on it and any subset $V_0 \subseteq V$ has a least upper bound and a greatest lower bound) and $\mathbb{F}$ is the set of all monotonic functions on V in any number of variables.

2. We assume $(V, \mathbb{F})$ satisfy the following. Let $\mathbf{h}_1, \ldots, \mathbf{h}_n$ be n functions in $x_1, \ldots, x_n$ (same n). Consider the vector function on V^n defined by

$$\vec{y} = \vec{\mathbf{h}}(\vec{x})$$

Where $\vec{x} = (x_1, \ldots, x_n), \vec{y} = (y_1, \ldots, y_n)$ and $\vec{\mathbf{h}} = (\mathbf{h}_1, \ldots, \mathbf{h}_n)$.

Then we assume that any such vector function for any n has at least one fixed point $\vec{x}_0$, i.e. we have

$$\vec{x}_0 = \vec{\mathbf{h}}(\vec{x}_0).$$

Note that the fixed point condition holds for $[0, 1]$ because of Brouwer's fixed point theorem and for the complete partial orders because of Tarksi's fixed point theorem.

Definition 3.2 (*Logics, models, theories and inconsistency*). Let $(V, \mathbb{F})$ be a function space.

1. By a $(V, \mathbb{F})$ logic we mean a system $(S, R, \mathbf{h}_t), t \in S$ where S is a finite set, $R \subseteq S^2$, and for each $t \in S$, the following holds:

Let $x_1, \ldots, x_n$ be all points in S such that tRx_i holds. Then $\mathbf{h}_t$ is an n-place function from $\mathbb{F}$.

It is important to note that when we write $\mathbf{h}_t$, then the variables $x_1, \ldots, x_n$ are ordered inside $\mathbf{h}_t$. We are not assuming that $\mathbf{h}_t$ is a symmetric function in its variables. Where does this order come from? It comes from the geometry of (S, R).

Consider for example Fig. 9, the points x', y, w are distinguishable individually by the geometry of the graph and so we can order them as we see fit when we plug them into $\mathbf{h}_z$.

It is expected that should the geometry not distinguish between any two points, say x_1 and x_2, then $\mathbf{h}_t$ must be symmetrical in these variables.

2. By a model for the logic $(S, R, \mathbf{h}_t)$ we mean any function $\mathbf{f}$ from S into V such that for all $t \in S$ we have

(*1) $\mathbf{f}(t) = \mathbf{h}_t(\mathbf{f}(x_1), \ldots, \mathbf{f}(x_n))$.

Such $\mathbf{f}$ s exist because of the fixed point theorem for $(V, \mathbb{F})$. $\mathbf{f}$ is not unique for solving (*1)

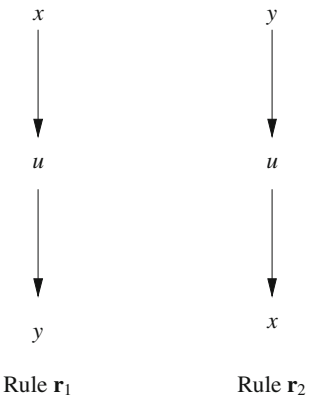


FIGURE 19.

3. Let $(S, R, \mathbf{h}_t)$ be a logic and let $s, t \in S$. We say $s \vDash t$ iff for any model $\mathbf{f}$ we have

(*2) $\mathbf{f}(s) = 1 \Longrightarrow \mathbf{f}(t) = 1$.

4. Note that (S, R) corresponds to the set of wffs of the language and $\mathbf{f}$ to truth value assignments to the wffs. The partial ordering of (1b) of Definition 3.1 above corresponds to the Lindenbaum algebra of the logic. The functions $\mathbf{h}_t$ are the truth tables of the connectives. R corresponds to the inductive construction tree of the wffs, except that it needs not the tree but could be a general binary dependence relation!

   This is a sort of "free style" notion of a wff.

   A wff is just an abstract point with a relation R saying xRy, which means y is an immediate subformula of x.

5. A theory in traditional logic is a set of wffs required to be true. In our context, to require a set of nodes to be true means to impose constraints on the models, i.e. on the function $\mathbf{f}$, i.e. on the solution of the equations of (*1).

   Since $\mathbf{f}$ is a general function from S into V, there are many types of constraints we can impose, e.g. that $\mathbf{f}$ has a minimal number of 0 values or that it has a minimal number of $\frac{1}{2}$ values (this would yield circumscription minimal models in the right context and right formulation for the case of 0 values, or would yield semi-stable semantics for the case of argumentation and value $\frac{1}{2}$).

   So we define

(*3) A logical theory is a set of constraints on the solutions (models) $\mathbf{f}$ s. It is inconsistent if the constraints have no solution.

   See [24] for examples.

Example (Geometrical proof rules, geometric inconsistency).

1. Let (S, R) be an argumentation frame. A unary geometrical proof rule $\mathbf{r}$ is a finite ordering (which can be represented visually by a figure) of

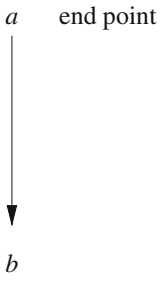


FIGURE 20.

the form $\mathbf{r} = (P_{\mathbf{r}}, R_{\mathbf{r}}, x, y)$ where $P_{\mathbf{r}}$ is a set of nodes and $R_{\mathbf{r}}$ is a binary relation on $P_{\mathbf{r}}$, and x, y are two distinct points in $P_{\mathbf{r}}$. x is called the input point (the premise of the rule) and y is called the output point (the conclusion of the rule). We require that x and y be geometrically definable in (S, R) in terms of information in $(P_{\mathbf{r}}, R_{\mathbf{r}})$. Figure 19 shows two such rules corresponding to the rules

$$\mathbf{r}_1 : \frac{A}{\neg\neg A}$$

And

$$\mathbf{r}_2 : \frac{\neg\neg A}{A}$$

2. A binary geometrical proof rule for (S, R) has the form $\mathbf{r} = (P_{\mathbf{r}}, R_{\mathbf{r}}, x, z', y)$ where $(P_{\mathbf{r}}, R_{\mathbf{r}})$ is finite ordering and $x, z', y \in P_{\mathbf{r}}$. x is the input point, z' is the base point and y is the output point. We require that x, y, z' be geometrically identifiable in (S, R) in terms of information available in $(P_{\mathbf{r}}, R_{\mathbf{r}})$.
3. What do we mean by a point being geometrically identifiable in (S, R) in terms of information available in $(P_{\mathbf{r}}, R_{\mathbf{r}})$? Consider Fig. 20.

   This pattern requires in (S, R) two points; the first being an endpoint attacking the second. The information $a =$ endpoint is for (S, R).
4. What do we mean by geometric inconsistency? We need to have a family of subsets of the network S which are marked unacceptable. Intuitively each such subset corresponds to a constraint of, say, wanting an equational solution which gives all members of the subset value 1. So the family is of all subsets for which there is no solution.

   A formal definition is given in Definition 3.3.

   Another example of a pattern for geometrical modus ponens is Fig. 7. The points x, y, z' in this figure are definable using $R_{\mathbf{r}}$.

Definition 3.3 (*Formal definition of geometric proof rules and inconsistency*).

1. Let (S, R) be an argumentation network with $R \subseteq S \times S$. Consider first order predicate logic with a binary relation R. Let $\Psi(x)$ be a formula in this language with a free variable x. Let $a \in S$. We can ask

$$(S, R) \models ?\Psi(a)$$

For example

$$\Psi_1(x) = \neg\exists y(xRy)$$

Says that x is an endpoint in (S, R) or

$$\Psi_2(x) = \exists y(xRy \wedge yRy)$$

We call such formulas $\Psi(x)$ as additional information about node x formulated in the first order predicate logic of the binary relation R.

2. A geometric rule $\mathbf{r}$ has the form

$$\mathbf{r} = (P_{\mathbf{r}}, R_{\mathbf{r}}, \Psi_t(x), a_1, \ldots, a_k, c, b), \text{ for } t \text{ in } P_{\mathbf{r}}$$

Where $a, a_i, c, b \in P_{\mathbf{r}}$.

$a_1, \ldots, a_k$ are inputs. c is the base and b is the output.

Intuitively think of it as $a_1 \wedge \ldots \wedge a_k \Longrightarrow_c b$.

$(P_{\mathbf{r}}, R_{\mathbf{r}})$ is a finite ordering and for each $t \in P_{\mathbf{r}}, \Psi_t(t)$ is additional information about t in the sense of item (1) above.

3. Let $\mathbf{r}$ be a geometric rule, and let (S, R) be an argumentation network. Let $\mu : P_{\mathbf{r}} \mapsto S$ be a one to one embedding of $P_{\mathbf{r}}$ into S. We say μ identifies the pattern of the rule $\mathbf{r}$ in $\mu(P_{\mathbf{r}})$ iff the following holds.
   (a) $xR_{\mathbf{r}}y$ iff $\mu(x)R\mu(y)$ for all $x, y \in P_{\mathbf{r}}$.
   (b) For all $x \in P_{\mathbf{r}}, (S, R) \vDash \Psi_x(\mu(x))$.
4. We require also that the points $\mu(a_1), \ldots, \mu(a_k)\mu(c)$ and $\mu(b)$ are uniquely identifiable in (S, R). More precisely there are properties $\varphi_{a_1}(x), \ldots, \varphi_{a_k}(x), \varphi_c(x), \varphi_b(x)$ such that for each $y', y \in \{a_1, \ldots, a_k, c, b)$ we have

$$(S, R) \vDash \varphi_y(\mu(y)) \wedge \bigwedge_{y' \neq y} \neg\varphi_{y'}(\mu(y)).$$

5. An inconsistency notion is just a family of subsets of S. For monotonic logic the family is closed under enlargement

Example. Consider the pattern of Fig. 20.

Here

$$\Psi_a(x) = \neg\exists yxRy$$
$$\Psi_b(x) = x = x.$$

Consider the following embeddings of this pattern into Fig. 9. (I chose this figure at random, just for illustration).

If we match (a, b) with (y, z), we get a good embedding because y satisfies Ψ_a, but if we embed (a, b) as (x', z), then the conditions are not satisfied.

Note that from now on to simplify notation we regard μ as the identity and talk about $P_{\mathbf{r}} \subseteq S$ and $R_{\mathbf{r}} \subseteq R$.

Definition 3.4 (*Geometrical proofs*). Let (S, R) be part of a logic as defined in Definition 3.2. Let $T \subseteq S$ be any subset. Let $\mathbf{r}_1, \ldots, \mathbf{r}_k$ be proof rules. We define the notion of

- the sequence $(x_1, \ldots, x_n), n \geq 1, x_i \in S$ is a proof of level $m \geq 0$ of x_n from T, using $\mathbf{r}_1, \ldots, \mathbf{r}_k$.

The definition is by induction on m and n.

Case $n = 1, m = 0$
x_1 is a proof from T if $x_1 \in T$.

Case $n + 1, m = 0$
$(x_1, \ldots, x_{n+1})$ is a proof from T iff one of the following holds:

1. $x_{n+1} \in T$
2. x_{n+1} is obtained from some $x_i, i \leq n$ using a unary geometrical rule $\mathbf{r} = (P_\mathbf{r}, R_\mathbf{r}, x, y)$ such that $P_\mathbf{r} \subseteq S, R_\mathbf{r} \subseteq R$ and x_i is the input $(x = x_i)$ and x_n is the output $(y = x_n)$.
3. For some $x_{j_1}, \ldots, x_{j_k}, x_j, i, j \leq n$ and some geometrical rule $\mathbf{r} = (P_\mathbf{r}, R_\mathbf{r}, x_1, \ldots,$
$x_k, z', y)$ we have $P_\mathbf{r} \subseteq S, R_\mathbf{r} \subseteq R, x_{j_i} = x_i, z' = x_j$ and $y = x_{n+1}$.
Case level $m + 1$
Assume that for each T and any $m' \leq m$ and any n we have defined the notion of $x_1, \ldots, x_n$ is a proof of x_n of level $\leq m'$. We now define this notion for level $m + 1$ and $n \geq 1$. Let cases (1)–(3) be as above for level $\leq m$. We add more cases
4. **Case** $m + 1, n = 1$
For some rule $\mathbf{r} = (P_\mathbf{r}, R_\mathbf{r}, x'_j, z', y)$ we have that there exists $(y_1, \ldots, y_{n'})$, $y_{n'} = y$ which is a proof of level $\leq m$ of y from $T \cup \{x'_j\}$. We also have $z' = x_1$.
5. **Case** $m + 1, n > 1$
For some rule as in (4), we have that there exists $(y_1, \ldots, y_{n'})y_{n'} = y$, which is a proof of level $\leq m$ of y from $T \cup \{x'_j\} \cup \{x_1, \ldots, x_{n-1}\}$. We also have $z' = x_n$.

Remark 3.5. Note that in logic based argumentation networks (see [14] or [29]) only level 0 proofs are used. The rules have the form $A_1 \wedge \ldots \wedge A_n \Longrightarrow_c B$ and only $\Longrightarrow_c$ eliminations are used.

Definition 3.6 (*Soundness of rules*). Let $(S, R, \mathbf{h}_t)$ be a logic in the sense of Definition 3.2. Let $\mathbf{r}_1, \ldots, \mathbf{r}_k$ be rules in the sense of Definition 3.3. We say the rules are sound iff whenever b is proved from a in (S, R) as in Definition 3.4 for $a, b \in S$ then $a \vdash b$ holds as defined in Definition 3.2.

We say the rules are complete iff we have

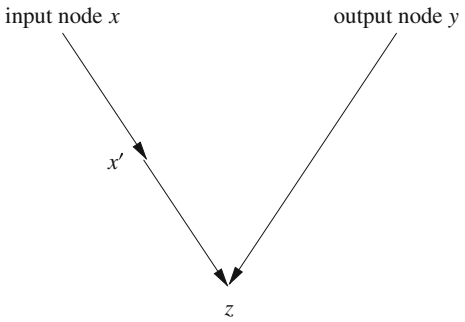
- $a \vdash b$ iff b is provable from a using the rules.

If a subset of S is marked inconsistent then the constraint arising from that set cannot be solved.

Example (Defeasible rules). Ordinary implication (strict implication) we can write as $A_1 \wedge \ldots \wedge A_n \rightarrow B$ or as $A_1 \rightarrow (A_2 \rightarrow \ldots \rightarrow (A_n \rightarrow B) \ldots)$.

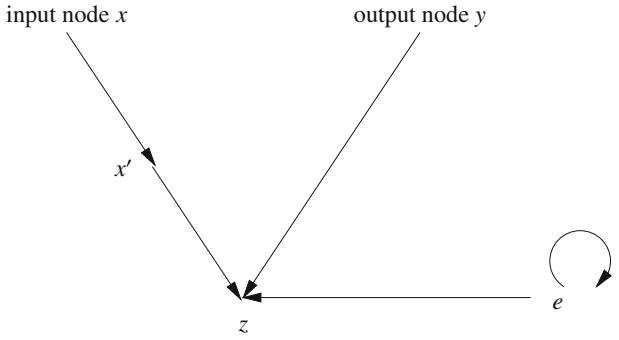
Let us do geometrically $A \rightarrow B$ and $A \Longrightarrow B$. All we need are some markers in the figures representing these two implications, to distinguish one from another. See Figs. 21 and 22.

Consider now (S, R) of Fig. 23 and consider the rule $\mathbf{r}$ of Fig. 22. Take the theory $T = \{a_1, \ldots, a_n\} \cup \{b_n\}$. What can it prove using $\mathbf{r}$?



$z = x \rightarrow y$, base node x' is an auxiliary node so that we can tell the difference between input and output. When identifying this pattern in an argumentation network it is required that nodes z and x' bear exactly the attacks shown in the pattern

FIGURE 21.



$z = x \Longrightarrow y$, base node x' is an auxiliary node. When identifying this pattern in an argumentation network it is required that nodes z, e and x' bear exactly the attacks shown in the pattern

FIGURE 22.

The answer is that it can prove b_0 and all $b_{n-1}, \ldots, b_1$ along the way. The deduction is essentially a_j and $b_j = [a_j \Longrightarrow (a_{j-1} \Longrightarrow \cdots (a_1 \Longrightarrow b_0) \cdots)]$ yields for $b_{j-1} = [a_{j-1} \Longrightarrow \cdots \Longrightarrow (a_1 \Longrightarrow b_0)]$.

We can turn this ordering into a logic if we give the functions $\mathbf{h}_t$, for any t of Fig. 23. Try $\mathbf{h}_{e_i} = \frac{1}{2}$. $\mathbf{h}_{a_i}$ = arbitrary $\mathbf{h}_{a'_i} = \mathbf{h}_{a_i}$. $\mathbf{h}_{b_0}$ = arbitrary. $\mathbf{h}_{b_j} = \min(1, 1 - a'_j + b_{j-1})$ for $j \geq 1$.

We need to show the rule is sound in this semantics, but in this case it is clear because the rules are versions of modus ponens and the function $\mathbf{h}_x$ are from Łukasiewicz many valued logic.

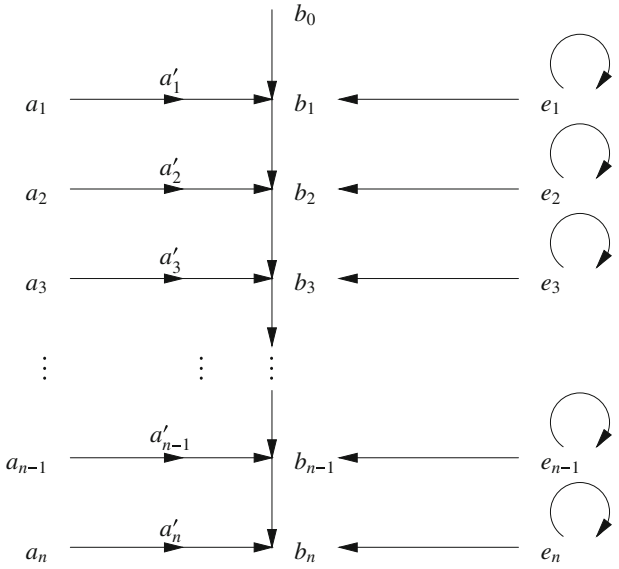


FIGURE 23.

3.2. Revisiting the Canonical Model

In Definition 2.6 we introduced a canonical ordering (Θ^*, R^*) made up of all Dagger wffs as well as an additional atoms Q^*.

Let us now treat it as a logic in the sense of Sect. 3.1.

Take $V = \{0, \frac{1}{2}, 1\}$ and

- $\mathbf{h}_t(x_1, \ldots, x_n) = 1 - \max(x_i)$

Let the relation $A \approx B$ be defined as saying

- $A \approx B$ for $A, B \in \Theta, A, B$ wff iff A and B have the same classical truth table.

This relation is an equivalence relation and is decidable.

Let $\Theta_0 = \Theta/\approx$

Let T be the set of all $\approx$ equivalence classes of Θ^*. Define an attack relation **Att** on T by $A/\approx$ **Att** $B/\approx$ iff for some $A' \approx A$ and $B' \approx B$ we have $B'RA$.

We claim[6]

- $A/\approx$ **Att** $B/\approx$ iff $A \vdash \neg B$ in classical logic.

What we need to show is that if $A \vdash \neg B$ then for some $A' \approx A$ and $B' \approx B$, A' is an immediate subformula of B'. This is not difficult to show. We can assume A and B are written in the language with $\Downarrow$ only.

Since $A \vdash \neg B$, we have $B \vdash \neg A$, hence $B \approx B \wedge \neg A$.

[6] This is a standard way of transferring any relation between points in a set to their equivalence classes.

Consider

$$\Downarrow (\Downarrow B, A)$$

This is equivalent to B and indeed A is an immediate subformula of it.

We thus got a canonical model of $(T,\ \mathbf{Att})$ of classical formulas (up to equivalence).

The above notion of Att is symmetric and is based on inconsistency in classical logic. In the literature, There are a number of frameworks for modelling argumentation in logic. A common assumption for logic-based argumentation is to start with a knowledge base **KB** in some logic (usually a version of defeasible logic based on **KB**) and construct arguments as pairs (Δ, A) where Δ is a minimal subset of the knowledge base such that Δ is consistent and $\Delta|\sim A$. Hunter [29] call the logic used for consistency and entailment, the base logic. Different base logics provide different definitions for consistency and entailment and hence give us different options for argumentation.

3.3. Argumentation Systems Arising from Knowledge Bases

We are going to use our results to analyse problems and abnormalities existing in argumentation systems based on knowledge bases formulated in some base logic see [13,14,29,34].

We begin with a specific paradoxical example, which according to Caminada and Amgoud [14] seems to defy solution.

Example. This is example 6 of [14, p. 293].

Our base logic has both strict rules with $\rightarrow$ and defeasible rules with $\Longrightarrow$.

Our knowledge base has the following data:

Strict data

1. a
2. d
3. g
4. $b \wedge c \wedge e \wedge f \rightarrow \neg g$

Defeasible data

7. $a \Longrightarrow b$
8. $b \Longrightarrow c$
9. $d \Longrightarrow e$
10. $e \Longrightarrow f$

From the above knowledge base we construct arguments. We follow [14] but use our notation to indicate the chain of reasoning.

Arguments

$$A = [a, a \Longrightarrow b]$$
$$B = [d, d \Longrightarrow e]$$
$$C = [a, a \Longrightarrow b, b \Longrightarrow c]$$
$$D = [d, d \rightarrow e, e \Longrightarrow f]$$

The problem with this example is that A, B, C, D have no defeaters and so one has to accept them and their conclusions which is the set $\{b, c, e, f\}$ is

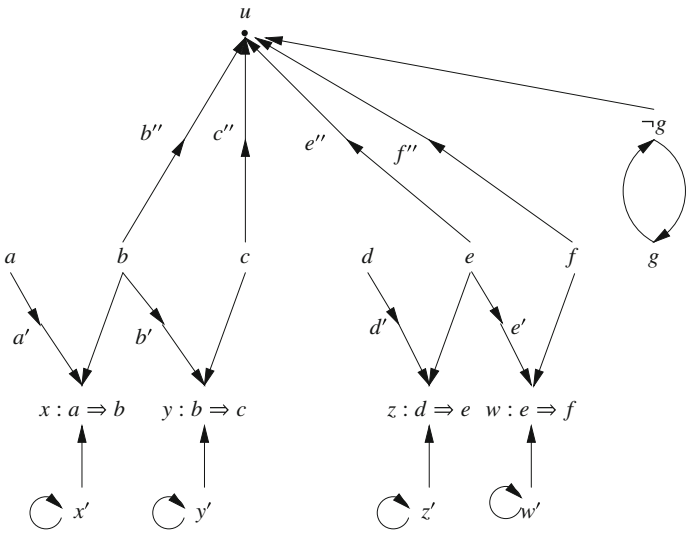


FIGURE 24.

justified which together with the data $\{a, g\}$ should be consistent but it is not consistent because of item 4 of the data.

Example (Representing Example 6 of [14] in our system). Figure 24 represents Example 6 of [14] as described above in our Example 3.3. The nodes $a', b', b'', c'', e', e'', f'', x', y', z', w'$ are auxiliary nodes. The original nodes mentioned are

$$a, b, c, d, e, f, g, \neg g$$

and

$$\begin{aligned} x &= (a \Longrightarrow b) \\ y &= (b \Longrightarrow c) \\ z &= (d \Longrightarrow e) \\ w &= (e \Longrightarrow f) \\ u &= b \wedge c \wedge e \wedge f \rightarrow \neg g. \end{aligned}$$

The rules of modus ponens are represented as in Figs. 21 and 22. This is why we need the auxiliary points. The arguments represent walks along Fig. 24, which respect the geometry of the rules.

They are:
$A =$ walk (a, b)
$B =$ walk (d, e)
$C =$ walk(a, b, c)
$D =$ walk(d, e, f)

We perceive the walk as a proof augmenting the knowledge base (which is a set of nodes), with additional nodes. So if **KB** is the original knowledge

base, then we perceive the arguments as follows:
A as $\mathbf{KB} \cup \{a, b\}$
B as $\mathbf{KB} \cup \{d, e\}$
C as $\mathbf{KB} \cup \{a, b, c\}$
D as $\mathbf{KB} \cup \{d, e, f\}$.

The perceptive reader might say: OK, fine. So you represented the problem your way. How are you going to solve the anomaly? A new representation does not solve any original problem! So?

The answer is that the new representation may suggest more readily a new idea for a solution to the problem. Once we get the new idea we can apply it equally well to the old original representation.

OK, so what is the new idea?

I say that what is wrong (in my opinion) with the old way of deriving arguments from a knowledge base is in the way they define attacks between arguments.

The concept is flawed and this is the source of difficulty. Let us recall our paper [26] entitled Logical modes of attack in argumentation networks. We presented there a model of attack between nonmonotonic knowledge bases, say Δ_1 and Δ_2. Δ_1 can attack Δ_2 by sending one or more items of data from itself (i.e. Δ_1) into Δ_2. This may render Δ_2 inconsistent or may hinder Δ_2 in its nonmonotonic deductive tasks. See [26, Example 1.8 Directional attacks].

How are we going to apply this definition in our case?

We saw that an argument becomes a walk along a graph. During the walk we collect points into our set. The more points we have collected, the more geometric patterns we can use to collect even more points. So imagine two people walking along a graph (say the streets of the old district in town). We first notice that the two fellows may not be using the same tourist guide (or map). So they may not have the same walks. In geometrical proof terms maybe the first fellow can use rule **r** while the second fellow does not use it. OK, now we ask: how can one obstruct the walk of the other?

1. One can be an immediate obstruction. Stop the other immediately in his tracks. This corresponds to a rebuttal (you should stop now or needed to stop before) or undercut (you got here by going through a "forbidden to walk" path!).
2. Give the guy a wrong direction so he will reach a dead end and will have to stop. How is this done in practice?

Remember each of these walkers is collecting nodes in the graph. If one of them gives the other some nodes then the enlarged set of the other can be inconsistent, or unacceptable (this is case (1) above), or he may be tempted to use the additional nodes to carry on walking and get to a pending inconsistency, (this is case (2) above).

Recall inconsistency was defined in Definition 3.3.

It is (2) above that is missing from the concept of attack. I hit upon a similar problem in the early 1990s when I was writing my book on Labelled Deductive Systems [19]. The notion was labelled revision theory and inconsistency. I could have a labelled theory which receives an input and really

eventually becomes inconsistent but it was not seen immediately. You needed to carry on proving all kinds of things from it to eventually realise it is inconsistent. So how are we going to use this idea in our context?

Let $(S, \breve{R})$ be an argumentation network with geometrical rules $\mathbf{r}_1, \ldots, \mathbf{r}_k$. Let $T \subseteq S$. This network is our base logic. We are using the fact that we can view a logic as network as we have shown in Sect. 3.1.

$(S, \breve{R}, T, \mathbf{r}_1, \ldots, \mathbf{r}_k)$ is our knowledge base **KB**, presented as a network. An argument X is a walk along this network. Let $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_m\}$ be two arguments. We say that Y attacks X if $T \cup \{x_1, \ldots, x_n, y_1, \ldots, y_m\}$ is not acceptable or not consistent according to the rules of X.

We may not see this immediately so we can talk about levels of pending inconsistency, inconsistency revealed after so many steps.

Note that the relation of attack is not symmetrical. X may not be able to consistently continue according to his rules but Y's rules may allow him to continue.

In our example the situation is symmetrical and C and D attack each other.

4. Predicate Argumentation

The discussions of previous sections allow us to put forward argumentation theory where the arguments are not atomic but are involved in predication, either by having parameters themselves, i.e. the arguments themselves are like predicates in predicate logic, or by being predicated upon, being themselves the elements of a meta-predicate.

Two immediate examples come to mind

1. The argument x itself involves a claim about certain domains, e.g.

   $$x = \text{all men are mortal.}$$

   See [6] for examples and references.
2. There is a predicate Q, a value predicate, operating on the argument x to form $Q(x, e)$,

   $$Q(x, e) = \text{The value of } x \text{ is } e,$$

   like that introduced by Bench Capon, [3,4], and there is a preference relation on values.

In this case the argument itself is being predicated upon by the value predicate.

This section will address predication and arguments.

The basic situation is shown in Fig. 25.

In this figure a is an element of the domain. $P(a)$ is a predicate about this element, say 'a is red'.

The argument $\forall x \neg P(x)$ is a sweeping general statement saying that nothing is red. The metapredicates Q and Q' tell us what source the statements come from. So $Q(P(a))$ says $P(a)$ comes from source Q and $Q'(\forall x \neg P(x))$ tells us that $\forall x \neg P(x)$ comes from source Q'. We consider Q' less reliable than Q. We can write $Q' < Q$.

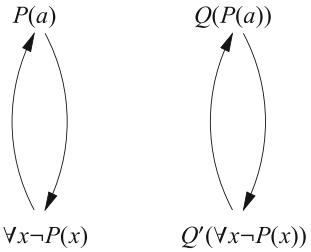


FIGURE 25.

In fact, the source might be a chain of hearsay. Q'' says he heard from Q' that .... So we might be comparing

$$Q_1 Q_2 \dots Q_k P(a)$$

with

$$Q'_1 Q'_2 \dots Q'_{k'} \forall x \neg P(x).$$

So really we need to handle chains of predicates $\alpha = (Q_1, \dots, Q_n)$. In real situations there may be several such chains as different witnesses tell stories which support the argument. See [16] for some examples and discussion.

Anyway, in our case the attack from $\forall x \neg P(x)$ on $P(a)$ cannot be accepted.

The figure has two features at the same time

1. Bench-Capon type value predicates Q, Q'.
2. The arguments themselves are predicate logic arguments.

We therefore need a language which allows us to express the following

1. Given a wff φ and a predicate $Q(x)$, we are allowed to write $Q(\varphi)$. (Think of Q as a provability predicate, for example.)
2. Try and make quantified $\forall x \varphi(x)$ behave like some instantiated $\varphi(x_0)$, in other words, eliminate quantification. Quantification can be eliminated if the domain is finite and known. Say $D = \{a_1, \dots, a_n\}$ then we can write

$$\exists x \varphi(x) = \bigvee_i \varphi(a_i)$$

$$\forall x \varphi(x) = \bigwedge_i \varphi(a_i)$$

But if we do not make such an assumption we still need a solution to how to eliminate quantifiers.

If we succeed in the above two tasks we will have reduced the problem of predicate argumentation to ordinary Dung networks. How are we going to do it?

There is a language introduced by Gabbay in [19,20], called **HFP**, Hereditarily Finite Predicates. It allows to write expressions like $P(\varphi)$.

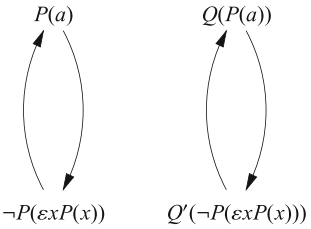


FIGURE 26.

It was studied in [20] in the chapter entitled "Self fibring of predicate logics". The language is powerful and has been applied to the logics of security among other applications.

To eliminate quantifiers we use Hilbert Epsilon symbol see [30].

We can write $\varepsilon x\varphi(x)$. This is a term which picks up in the model an element satisfying φ and if no such element exists (i.e. $\forall x\neg\varphi(x)$ holds) then it picks up an arbitrary element. Thus we have

$$\varphi(\varepsilon x\varphi(x)) \to \exists y\varphi(y)$$
$$\neg\varphi(\varepsilon x\varphi(x)) \to \forall y\neg\varphi(y)$$

So by using the Epsilon symbol, we don't need quantifiers and the language can be treated like it were propositional.

So Fig. 25 becomes Fig. 26.

This is a propositional network. We need to somehow extract from it by geometric means the reason for the attack relation and why the Q' cannot attack the Q. Let us begin first by taking predicate logic formulated with the ε-symbols. We choose monadic logic for simplicity. We have a set $\mathbb{P}$ of unary predicates, variables V and the connective $\Downarrow$ and the ε-symbol.

Definition 4.1. Formulas and terms are defined as follows

1. x is a term with x free, for $x \in V$.
2. If $\mathbf{t}(x_1,\ldots,x_n)$ is a term with x_i free then $P(\mathbf{t}(x_1,\ldots,x_n))$ is a formula with x_i free.
3. If $\varphi(x,x_1,\ldots,x_n)$ is a formula with x,x_i free then $\varepsilon x\varphi(x)$ is a term with x_i free.
4. If A_i are wffs with $\{x^i_j\}$ for each $i = 1,2,\ldots$ then $\Downarrow\{A_i\}$ is a wff with $\{x^i_j\}$ free.

Definition 4.2. Now that we have the wffs of the language we can define a canonical network as follows.

Let S^* be the set of all wffs together with the new additional predicates $P^*(x)$, for any predicate P and variable x.

Define R^* on S^* by

1. $P(x)R^*P^*(x)$ and $P^*(x)R^*P(x)$.
   This gives us an assignment
2. AR^*B if B is an immediate subformula of A
   This gives us the table for $\Downarrow$

3. $\Downarrow \varphi(\varepsilon x\varphi(x))R^*\varphi(y), y$ any term
   This corresponds to the quantifier rule $\forall x\neg\varphi(x) \to \neg\varphi(y)$.

Remark 4.3. 1. Note that we do not have the axiom:

$$(\sharp) \qquad \forall x(P(x) \leftrightarrow Q(x)) \to (\varepsilon xP(x) = \varepsilon xQ(x)).$$

This means that $\varepsilon xP(x)$ and $\varepsilon x(P(x) \wedge P(x))$ may choose different elements. So our semantics for the logic without $(\sharp)$ is syntactic. A model is a function $\mathbf{m}$, giving values to all wffs such that

$$(*) \qquad \mathbf{m}(P(y) = 1 \Longrightarrow \mathbf{m}(P(\varepsilon xP(x))) = 1$$

If we do adopt $(\sharp)$ as an axiom, we can have set theoretic models, where we have a domain D and a selection function $\mathbf{s}$ giving selection from any subset $D_0 \subseteq D$ and element $\mathbf{s}(D_0) \in D_0$. If $D_0 = \varnothing$, then $\mathbf{s}(\varnothing) \in D$.

We now have

$$\varepsilon xP(x) = \mathbf{s}\{x \mid P(x) \text{ holds}\}.$$

2. The axiom $(\sharp)$ is written with $\to$, the universal quantifier and $=$. The language we use has $\Downarrow$, the Epsilon symbol and no equality. We can express $\to$ using $\Downarrow$ and express the universal quantifier using the Epsilon symbol. To overcome the lack of equality we can write the axiom as

$$(\sharp 1) \qquad \forall x(P(x) \leftrightarrow Q(x)) \to (A(\varepsilon xP(x)) \to A(\varepsilon xQ(x)))$$

Where A is an arbitrary new predicate. This expression is OK because the axiom has implicit quantifier $\forall$, i.e., it is

$$\forall P\forall Q\forall A(\sharp 1).$$

Proposition 4.4. *The stable extensions of the canonical network (S^*, R^*) of Definition 4.2, yield exactly all predicate models with ε-symbols based on $\Downarrow$ and V.*

Proof. Let $\mathbf{m}$ be a model for the language. Following Remark 4.3, $\mathbf{m}$ is a syntactical model, giving values in $\{0, 1\}$ to all wffs, and satisfying

$$\mathbf{m}(\varphi(y)) = 1 \to \mathbf{m}(\varphi(\varepsilon x\varphi(x))) = 1.$$

Consider now the extension $\mathbb{E}_{\mathbf{m}}$ defined by

$$P(x) \in \mathbb{E}_{\mathbf{m}} \quad \text{if } \mathbf{m}((x)) = 1$$
$$P^*(x) \in \mathbb{E}_{\mathbf{m}} \quad \text{if } \mathbf{m}(_(x)) = 0.$$

We now check an arbitrary point $A \in S$, that A is in $\mathbb{E}_{\mathbf{m}}$ iff $\mathbf{m}(A) = 1$. This we do by structural induction which is also induction on the tree (S, R).

1. A is attacked by all its main subformulas Y. If $A =\Downarrow \{Y\}$, then really $A = \bigwedge_Y \neg Y$ and so if $\mathbf{m}(Y) = 1$, which by the induction hypothesis means that Y is in the extension $\mathbb{E}$, for any Y, then $\mathbf{m}(A) = 0$, and also A is not in the extension $\mathbb{E}$. If for all $Y, \mathbf{m}(Y) = 0$, then by the induction hypothesis, no such Y is in $\mathbb{E}_{\mathbf{m}}$. Therefore A is in $\mathbb{E}_{\mathbf{m}}$, and we also have that $\mathbf{m}(A) = 1$, because $A = \bigwedge \neg Y$. Thus we conclude for this case that A is in $\mathbb{E}_{\mathbf{m}}$ iff $\mathbf{m}(A) = 1$.

2. We now check the case of $A =\Downarrow B$ where $B = \varepsilon x\varphi(x))$. In this case we have that A is attacked by B and also by any $\varphi(y)$, for any term y.

   We must show that if any of the attackers Z gets value 1 (i.e. $\mathbf{m}(Z) = 1$ which means by the induction hypothesis that Z is in $\mathbb{E}_{\mathbf{m}}$) then $\mathbf{m}(A) = 0$, and A is not in the extension $\mathbb{E}_{\mathbf{m}}$. Otherwise $\mathbf{m}(A) = 1$.

   If $\mathbf{m}(B) = 1$, then clearly $\mathbf{m}(A) = 0$. If $\mathbf{m}(\varphi(y)) = 1$, then by (*) $\mathbf{m}(B) = 1$ and so $\mathbf{m}(A) = 0$. If $\mathbf{m}(\varphi(y)) = 0$ for all y, then $\mathbf{m}(\neg B(\varepsilon xB(x))) = 1$, but also from the argumentation network point of view, $A \in \mathbb{E}_{\mathbf{m}}$.
3. Assume we have a stable extension $\mathbb{E}$. Let $\mathbf{m}_{\mathbb{E}}$ be defined by

$$\mathbf{m}_{\mathbb{E}}(P(x)) = 1 \quad \text{iff } P(x) \in \mathbb{E}.$$

   To show that this holds for any $A \in S$, we follow similar reasoning as in case (1). □

Let us now also deal with predicates on predicates of the form $Q(P(x))$. There are difficulties with the ε-symbol in this case, as we shall see. Let us give the formal definitions first and then follow with a discussion.

Definition 4.5 (*Syntax*). Consider a language with variables $V = \{x_1, x_2, \ldots\}$, the Dagger connective $\Downarrow$ and the ε-operator (εx) and a set of unary predicates

$$\mathbb{P} = \{P_1, P_2, \ldots\}.$$

We define the notion of a term and a formula with free variables

1. Any $x \in V$ is a term. x is free in the term x.
2. If P is an atomic predicate and $\mathbf{t}(x_1, \ldots, x_n)$ is a term with the free variables $x_1, \ldots, x_n$, then $P(\mathbf{t}(x_1, \ldots, x_n))$ is a formula with the free variables $x_1, \ldots, x_n$.
3. Let $\alpha = (P_1, \ldots, P_k)$ be a sequence of elements from $\mathbb{P}$. If $\varphi(x, x_1, \ldots, x_n)$ is a formula with the free variables $x, x_1, \ldots, x_n$, then $\varepsilon^{\alpha} x\varphi(x)$ is a term with the free variables $x_1, \ldots, x_n$.
4. If Q is a predicate and $\varphi(x_1, \ldots, x_n)$ is a formula with free variables $x_1, \ldots, x_n$ then $Q(\varphi)$ is a formula with free variables $x_1, \ldots, x_n$.
5. If $\varphi_i, i = 1, \ldots, k$ are formulas with free variables $x^i_j, i = 1, \ldots, k, j = 1, \ldots, n(i)$, respectively, then $\Downarrow \{\varphi_i\}$ is a formula with the free variables

$$\{x^i_j \mid i = 1, \ldots, k, j = 1, \ldots, n(i)\}$$

6. Let $\mathbb{L}(\mathbb{P})$ be the language defined by clauses (1), (2), (5). $\mathbb{L}(\mathbb{P}, \varepsilon)$ the language defined by (1), (2), (3), (5) and $\mathbb{L}(\mathbb{P}, \varepsilon, \text{Fib})$ be the language defined by (1)–(5).

Definition 4.6 (*Models*). Let $\mathbb{P}^*$ be the set of all finite sequences from $\mathbb{P}$. For each such sequence $\alpha \in \mathbb{P}^*$, let $\mathbf{m}_{\alpha}$ be a classical monadic model for the language $(\mathbb{P}, \Downarrow)$, i.e. $\mathbb{L}(\mathbb{P})$. We turn the system $\{\mathbf{m}_{\alpha}\}$ into a model of our syntax.

We can assume all models $\mathbf{m}_{\alpha}$ have domain V.

First we convert each $\mathbf{m}_{\alpha}$ into a model of the language $\mathbb{L}(\mathbb{P}, \varepsilon)$. We need to deal with $\varepsilon^{\alpha} x\varphi(x)$ and assign it a value. We use induction.

1. For a wff φ without ε, let $\varepsilon^{\alpha}x\varphi(x)$ be any x such that $\mathbf{m}_{\alpha} \models \varphi(x)$. Otherwise let $\varepsilon^{\alpha}x\varphi(x)$ be any element. Note that if $\mathbf{m}_{\alpha} \models \exists x\varphi(x)$ then $\varphi(\varepsilon^{\alpha}x\varphi x)$ holds, otherwise $\neg\varphi(\varepsilon^{\alpha}x\varphi(x)$ holds. We say that $\varepsilon^{\alpha}x\varphi(x)$ has been assigned a value.

   Also note, following Remark 4.3 that we are not committed to assigning the same element to the Epsilon symbol applied to two logically equivalent formulas.
2. Let $\Psi(x, \mathbf{t}_1, \ldots, t_k)$ be a wff of classical logic with ε and assume all $\varepsilon^{\beta}x\varphi(x)$ in Ψ have been assigned values in $\mathbf{m}_{\alpha}$ for any α, β. This means that Ψ itself can be evaluated in $\mathbf{m}_{\alpha}$, because all expressions $\varepsilon^{\beta}x\varphi(x)$ in Ψ are assigned elements in $\mathbf{m}_{\alpha}$. So let $\varepsilon^{\alpha}x\Psi(x)$ be some arbitrary element in $\mathbf{m}_{\alpha}$ which satisfies Ψ, otherwise, if there is no element in $\mathbf{m}_{\alpha}$ satisfying Ψ, choose any element.

Thus we now have that each $\mathbf{m}_{\alpha}$ can be considered a model of $\mathbb{L}(\mathbb{P}, \varepsilon)$. We now need to extend our definition to model expressions like $Q(\varphi)$. We regard Q as a modality

$$\mathbf{m}_{\alpha} \models Q(\varphi) \text{ iff } \alpha * (Q) \models \varphi$$

where $\alpha * (Q)$ is the sequence obtained from α by adding Q at the end.

For example, we have

$$\alpha \models P_1(P_2(P_3(\varphi))) \text{ iff } (\alpha, P_1, P_2, P_3) \models \varphi.$$

Remark 4.7. Note that now we have syntax and semantics. So let us see what theorems are valid.

1. $\alpha \models \neg\varphi(\varepsilon^{\alpha}x\varphi(x) \rightarrow \neg\varphi(y)$
   $\alpha \models \varphi(\varepsilon^{\beta}x\varphi(x)) \rightarrow \varphi(y)$
2. $\alpha \models Q(\Downarrow \{A, B\})$ iff $(\alpha, Q) \models\!\Downarrow (A, B)$
   iff $(\alpha, Q) \not\models A$ and $(\alpha, Q) \not\models B$
   iff $\alpha \not\models Q(A)$ and $\alpha \not\models Q(B)$
   iff $\alpha \models\!\Downarrow (Q(A), Q(B))$.

   We can see that the following holds

$$Q(\Downarrow \{A_i\}) \equiv\!\Downarrow \{Q(A_i)\}.$$

3. Note that although $\mathbf{m}_{\alpha}$ have the same domain V, the element which $\varepsilon^{\alpha}xP(x)$ picks up at each α may be different. The reason is that the extension of P at each $\mathbf{m}_{\alpha}$ may be different and so we will not be able to organise the same choice for $\varepsilon^{\alpha}xP(x)$.
4. Consider $\varepsilon^{\alpha}x(Q_1, Q_2P(x))$. We are choosing an x at α, such that $P(x)$ holds at $\beta = (\alpha, Q_1, Q_2)$. This is possible because all models have the same domain V and for each Q the "modality" suggested by Q is linear discrete. Generally in modal logic when we write $t \models \Box\exists xP(x)$ we may have a different c_s such that $s \models P(c_s)$ for each s such that tRs (s accessible to t). We have

$$t \models \Box P(\varepsilon xP(x))$$

to deduce

$$t \vDash \exists z \Box P(x)$$

but this is not true!

So the rule in general

$$\vdash \varphi(y) \rightarrow \exists z \varphi(z)$$

becomes problematic.

We avoid this problem because all of our ε-symbols have the form $\varepsilon^{\beta} x$ for some β. So $\varepsilon^{\beta} x P(x)$ goes to $\mathbf{m}_{\beta}$ only.

Let us illustrate: Assume

$$\mathbf{m}_{\alpha} \vDash Q_1(P(a)) \wedge Q_2(P(b))$$

Thus

$$\mathbf{m}_{\alpha} \vDash Q_2(P(\varepsilon^{(\alpha, Q_1)} x P(x)) \wedge Q_2(P(\varepsilon^{(\alpha, Q_2)} x P(x))).$$

We have no problems with that.

However, had we not indexed the ε operator we would have got

$$\mathbf{m}_{\alpha} \vDash Q_1(P(\varepsilon x P(x))) \wedge Q_2(P(\varepsilon x P(x)))$$

which would have allowed us to deduce

$$\mathbf{m}_{\alpha} \vDash \exists y [Q_1(P(y)) \wedge Q_2(P(y))]$$

which may not be true.

Another approach is to make elements of the domain be vectors of the form $y = (a, b)$ and the semantics would be different. Which solution we choose depends on the application. Our aim is not to solve difficulties with the ε-symbol but to apply it to argumentation. For this reason we also simplify by adding the axiom

$$Q \Downarrow \{A_i\} \equiv \Downarrow \{Q A_i\}.$$

Our purpose is not to develop a comprehensive theory of **HFP** $+\varepsilon$ symbol but to develop a predicate argumentation theory and so we need less of the **HFP** $+\varepsilon$-symbol.

Example. Let us revisit Fig. 26 and see how it can be dealt with in our semantics. This discussion is only intuitive at this stage. Consider Fig. 27. Remember that $\neg\varphi$ is $\Downarrow \varphi$ in the Dagger language.

We have $Q' < Q$ and α is related to (α, Q) and to (α, Q').

We would need to say that there cannot be any attack from world Q' to Q is $Q' < Q$.

We shall present a precise system later. We already see that we may need to allow attacks from one world α to another world β.

There is a way to simplify. We notice that argument $P(a)$ and argument $Q(P(a))$ are both atomic units and are independent of one another. So we can put them all in one network/model. We can recognise the world Q by taking all arguments of the form $Q(x)$. So we can have one network and the different worlds are subnetworks of it. This allows us to represent $Q' < Q$ by an attack from Q to Q' done in a certain way. The next Example 4 will illustrate.

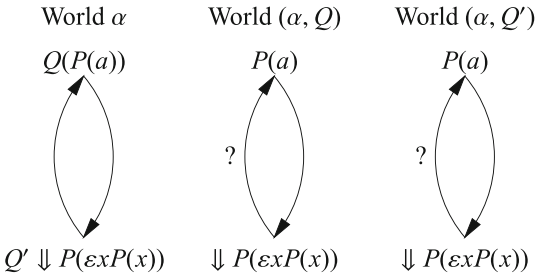


FIGURE 27.

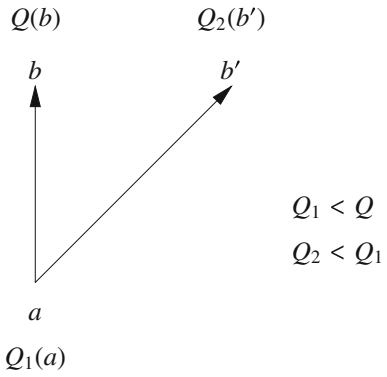


FIGURE 28.

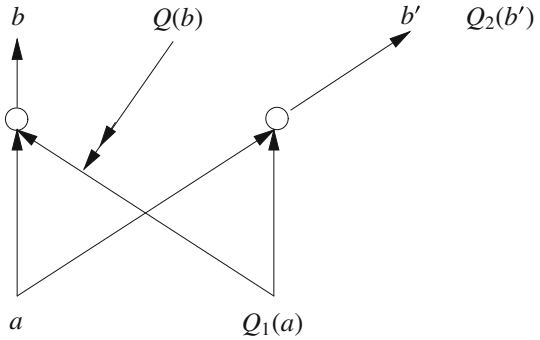


FIGURE 29.

Example. We consider the basic situation of Fig. 28 and implement it in Fig. 29.

We have that argument a attacks arguments b and b'. Argument a has metalevel value Q_1, argument b has value Q and argument b' has value Q_2.

The terms $Q_1(a), Q(b)$, and $Q_2(b')$ are not part of the attack network but are metalevel to it. The information $Q_1 < Q$ and $Q_2 < Q_1$ say which value is higher than which in the Bench-Capon sense [3].

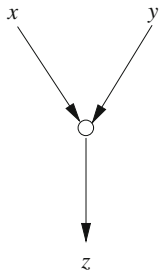
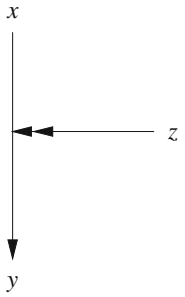


FIGURE 30.



z attacks the arrow from x to y

FIGURE 31.

Thus the attack of a on b cannot take place because b has higher value but the attack on b' is OK because a has higher value.

Now, how are we going to represent this metalevel information in the object level? We need:

1. To include $Q_1(a), Q(b)$ and $Q_2(b')$ as arguments in the object level. We know how to do that from the previous discussion in this section.
2. We need to represent the metalevel information $Q_1 < Q$ and $Q_2 < Q_1$ in the object level. To do this we need two higher level concepts introduced in our papers [22,23]. These are
   (a) joint attacks
   (b) higher level attacks.

   The concept of joint attacks is illustrated in Fig. 30 and is discussed in [22].

   x and y join forces to attack z. For the attack to succeed, we need both x and y to be "in" and to be joined.

   The concept of higher level attack [23], which originates in [1][7] allows arguments to attack and disconnect attacks arrows. This is illustrated in Fig. 31

[7] This concept was later independently discovered by Modgil [32] and studied by Baroni and others [8]. See discussion and references in [23].

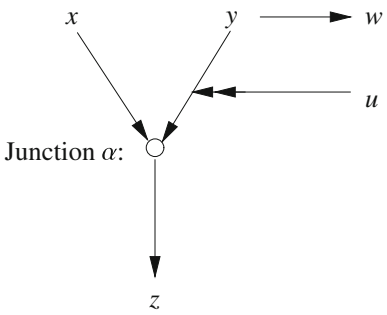


FIGURE 32.

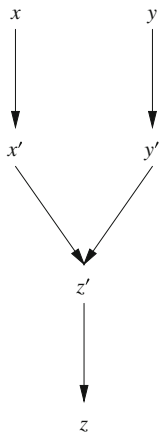


FIGURE 33.

If z is "in" the attack from x to y is "out" (x, y are still untouched).

Figure 29 represents our implementation of Fig. 28, where we use joint attacks and higher level attacks. We have that the double arrow emanating from $Q(b)$ onto the arrow which joins $Q_1(a)$ to the attack on b, is representing the fact that $Q_1 < Q$ and therefore making sure that a cannot attack b.

Note that in Fig. 32 the following holds:

u attacks the contribution of y to the joint attack (on z), which is forming, (or getting organised or "gathering") at junction α. This means that the joint attack cannot go on, if u is in. It does not mean that x can attack alone! This is why we have the junction notation.

y itself is not attacked by u, so y's attack on w is valid. In Fig. 32 we have the extension $u = \text{in}$, $y = \text{in}$, $w = \text{out}$, $x = \text{in}$, $z = \text{in}$.

Remark 4.8 (Representation of joint attacks, and higher level attacks). In Example 4 and in Fig. 29 we used joint attacks and higher level attacks to code the metalevel information that value Q_1 is lower than value Q_2, written as $Q_1 < Q_2$. We therefore need to show how to represent joint attacks

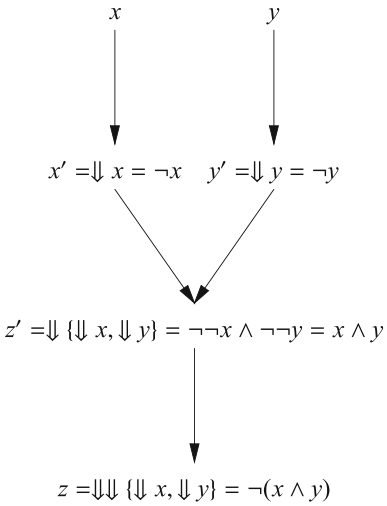


FIGURE 34.

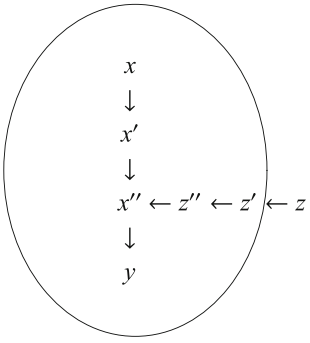


FIGURE 35.

and higher level attacks in our canonical model. We need to translate Figs. 30 and 31 into the canonical model.

We start with joint attacks. Following [22], consider Fig. 33.

Note that only when x and y are "in", do we get that z is out.

Consider now Fig. 34.

Clearly this is the same figure as Fig. 33, and we can see that the joint attack of x and y is executed by $x \wedge y$ in the canonical model.

Let us now address higher level attack. Consider Fig. 30. This can be implemented by Fig. 35 (which is the same as Figure 9 of [23, p. 365].

We note that this figure represents the higher level attack of Fig. 31, as shown in [23] and as can be readily verified directly. We further observe that the part of Fig. 35 enclosed in a circle, is the same as the representation of joint attacks in Fig. 34. This is a joint attack of x and z' on y. We note that z' is really $\neg z$. So we get that Fig. 31 can be represented as Fig. 36.

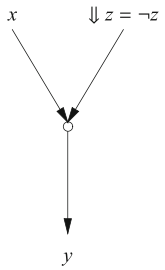


FIGURE 36.

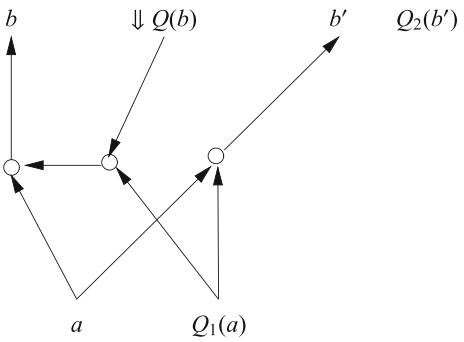


FIGURE 37.

Now since the case of joint attacks we already know how to represent (Fig. 34), we can also represent higher level attacks.

Remark 4.9. Following our discussion in Remark 4.8, Fig. 29 becomes Fig. 33. Now consider Fig. 37. In this figure the fact that $Q_1 < Q$ is represented by the fact that $\Downarrow Q(b)$, (i.e. $\neg Q(b)$) joins the contribution of $Q_1(a)$ to the joint attack of $\{a, Q_1(a)\}$ on b. We thus have the joint attack of $[a$ and the joint attack of $\{Q_1(a), \neg Q(b)\}]$ jointly attacking b. In symbols:

$$\text{Joint}\{a, \text{Joint}\{Q_1(a), \neg Q(b)\}\} \text{ attack on } b.$$

Note that what we have not done yet is to have a uniform representation of how $Q'(x)$ joins all the attacks of x on any y and how any $Q''(y)$ also participates but Q'' participation is effective only if $Q' < Q''$.

So far we have participation of Q'' with Q' only when $Q' < Q''$.

5. Resource Considerations

Let us do again Definition 1.5, this time paying attention to resources.

Definition 5.1. Let (S, R, α) be a decorated ordering. Let h be a model. Let us define the notion of resource annotated valuation of nodes in S. We write $\text{Val}(h, x) = (\mathbf{f}(x), \mathbf{F}(x))$.

1. $\mathrm{Val}(h,x) = (h(\alpha(x)), \{x\})$ for x an endpoint of the ordering.
2. $\mathrm{Val}(h,x) = (1 - \max_{xRy}(\mathbf{f}(y), \bigcup_{xRy} \mathbf{F}(y))$ for x which is not an endpoint.
3. For any $x, \mathbf{f}(x)$ is the truth value in $\{0,1\}$ of the formula $\alpha(x)$ and $\mathbf{F}(x)$ indicates which end nodes x relies on. $\mathbf{F}(x)$ is needed for resource logic, not for classical logic.

Example (Resource considerations). We now explain the components of Definition 5.1.

Consider the following deduction

1. A, assumption
2. $A \to (A \to B)$, assumption
3. $A \to B$, from (1) and (2) by modus ponens.
4. B from (1) and (3) by modus ponens.

The above deduction is not valid in linear logic because (1) A is used twice (in (3) and (4).

A correct deduction for linear logic would be the following

1. A, assumption
2. A, assumption
3. $A \to (A \to B)$, assumption
4. $A \to B$, from (2) and (3)
5. B, from (1) and (4).

The above means that the formula

$$C = (A \wedge [A \to (A \to B)] \wedge \neg B)$$

is not a contradiction in resource logic as described above. While

$$D = A \wedge A \wedge (A \to (A \to B)) \wedge \neg B$$

is a contradiction.

Let us now reflect the resource idea in our $\Downarrow$ system.

Example (Resources considerations for $\Downarrow$).

1. First recall a convention for drawing figures. When we have a binary relation $R \subseteq S \times S$ and we have xRy, we draw it as $x \leftarrow y$ (arrow going into the x).

   We can now build two possible decorated acyclic graphs for the formula $\neg A \wedge (\neg A \to (\neg A \to B)) \wedge \neg B$. We use the equivalent $\neg A \wedge (A \vee A \vee B) \wedge \neg B$. See Figs. 38 and 39.

In Fig. 38 all the elements of $\mathbf{F}(s)$ are different. No node is used twice. In Fig. 39 all nodes are used twice. In classical logic it does not matter. In resource logic it does matter.

Remark 5.2 (Argumentation frames based on linear logic). The connection between logic and networks, together with the discussion and figures of this section shows that for the case of acyclic networks, if we allow for each node in the network to attack only a single other node at the most, then the corresponding underlying logic is linear propositional logic based on Dagger, and the network is actually a tree.

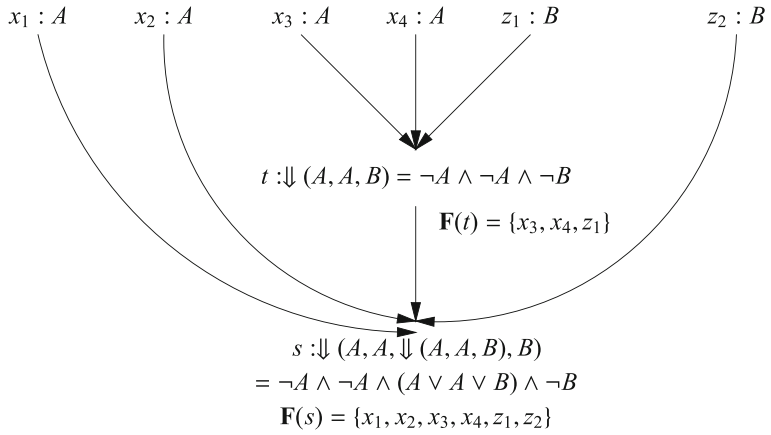


FIGURE 38.

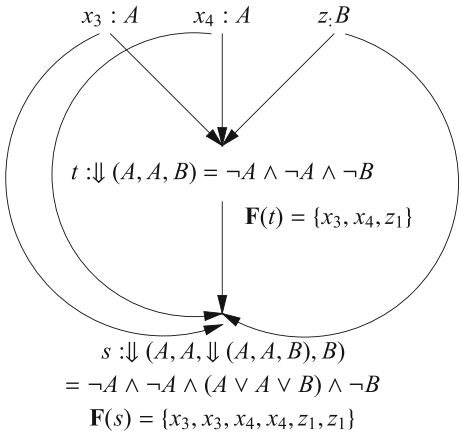


FIGURE 39.

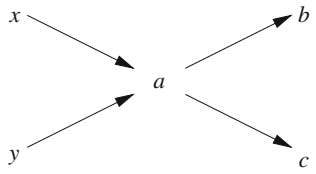


FIGURE 40.

If we want a node to attack several other nodes then we must make multiple copies of it. Thus for example Fig. 40 becomes Fig. 41.

If we allow cycles the definitions are more complicated.

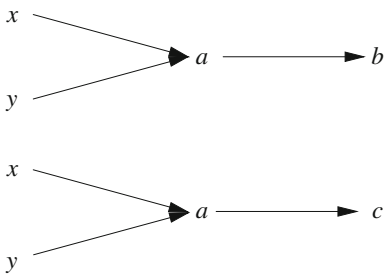


FIGURE 41.

6. Conclusion: Discussing the Correspondence Between Dung Networks and Dagger Logic and Comparing it with Seemingly Relevant Literature

In this concluding section, we would like to evaluate the correspondence between Argumentation networks and classical logic formulated with the Dagger connective. We try and answer possible criticisms from both the pure logic community and the down to earth practical argumentation community.

The title of this paper may be seen by some as a sweeping statement. We are saying that Dung argumentation frames together with its machinery of finding extensions is essentially equivalent to classical propositional logic with the Pierce-Quine Dagger connective and its machinery for finding models for wffs.

We have discussed and demonstrated the connections under (S1) and (S2) of Sect. 2.2. However, since there is a general confusion in the community about what it means to be an "equivalent" system, we thought we had better clarify these concepts in this section. Furthermore, this clarification will allow us to compare this work with some seemingly related works, namely, References [5,6,15,17,28,38].

Let us begin with a very simple example of two systems which are *not* the same. Take two strong enough computer languages. Say modern Basic and Pascal. These languages are strong enough for each to simulate the other. Give me a program in one and I can write an equivalent program in the other, where "equivalent" means doing exactly the same job!

Yet, despite the above, we will not say the two languages are the same. Yes, they have the same expressive power, Yes, they can do the same jobs. However, their basic internal constructs are different. They are not based on the same "internal movements".

We ask the reader to accept this intuitively. Don't ask us what we mean by "internal movements".

We now show you two systems which are generally considered the same.

Start with intuitionistic implication $\Longrightarrow$, formulated and axiomatised as a Hilbert System, call it **I**. We have the following two axioms and the rules of Modus ponens and substitution.

1. $a \Longrightarrow (b \Longrightarrow a)$
2. $(a \Longrightarrow (b \Longrightarrow c)) \Longrightarrow ((a \Longrightarrow b) \Longrightarrow ((a \Longrightarrow c))$

We now want to add negation to the system. We do it in two ways, creating two separate systems $\mathbf{I}_1$ and $\mathbf{I}_2$. $\mathbf{I}_1$ adds a negation unary symbol $\neg_1$ with the following additional axioms:

3. $(a \Longrightarrow \neg_1 b) \Longrightarrow (b \Longrightarrow \neg_1 a)$
4. $a \Longrightarrow (\neg_1 a \Longrightarrow b)$

The second system $\mathbf{I}_2$ adds a constant to the logic, call it $\mathbf{f}$, with the additional axiom

5. $\mathbf{f} \Longrightarrow a$

We now show that the two systems are really the same.

$\mathbf{I_2}$ finding itself inside $\mathbf{I_1}$

We are now in the realm of $\mathbf{I}_1$. We want to look at $\mathbf{I}_1$ through the eyes of $\mathbf{I}_2$. $\mathbf{I}_2$ has a constant $\mathbf{f}$. Can we find it in $\mathbf{I}_1$? Yes. Let $\mathbf{f}_1$ be any $q \wedge \neg_1 q$. We must prove in $\mathbf{I}_1$ that any two $p \wedge \neg_1 p$ and $q \wedge \neg_1 q$ are equivalent (i.e. from axioms (1)–(4) we prove $\mathbf{I}_1 \vdash (p \wedge \neg_1 p) \Longrightarrow (q \wedge \neg_1 q)$).

This makes $\mathbf{f}_1$ unique. We now prove in $\mathbf{I}_1$ that $\mathbf{f}_1 \Longrightarrow a$ is a theorem of $\mathbf{I}_1$. Having done all that we managed to look at $\mathbf{I}_1$ through the eyes of $\mathbf{I}_2$. We found an $\mathbf{f}_1$ which is what $\mathbf{I}_2$ has.

$\mathbf{I_1}$ finding itself in $\mathbf{I_2}$

Here we want to look at $\mathbf{I}_2$ through the eyes of $\mathbf{I}_1$. $\mathbf{I}_2$ has $\neg_1$ as negation. Can we find it in $\mathbf{I}_2$? The answer is yes. Let

$$\neg_2 a = a \Longrightarrow \mathbf{f}.$$

We now need to prove in $\mathbf{I}_2$ (using axioms (1), (2) and (5)) that axioms (3) and (4) hold for $\neg_2$.

This means prove in $\mathbf{I}_2$ that:

3*. $(a \Longrightarrow (b \Longrightarrow \mathbf{f})) \Longrightarrow (b \Longrightarrow (a \Longrightarrow \mathbf{f}))$
4*. $a \Longrightarrow ((a \Longrightarrow \mathbf{f}) \Longrightarrow b)$

This must be done, and can be done, using axioms (1), (2), (5) of $\mathbf{I}_2$.

Why is it that all logicians agree that $\mathbf{I}_1$ and $\mathbf{I}_2$ are the same really? It is because both formulations of negation are based on the same idea. As it says in the Bible, "Keep away from falsity".

Now let us compare the above with what we did in (S1) and (S2) of Sect. 2.2. The Peirce–Quine connective dagger of logic and the Dung attack relation of argumentation are based on the same idea. This we have shown. Now the question is are we doing for logic and argumentation network something similar to what we did for $\mathbf{I}_1$ and $\mathbf{I}_2$?

Logic finding itself in the argumentation world. The typical figure for logic is Fig. 3. As discussed in (S2), of Sect. 2.2, logic can "find" this figure in the argumentation world as Fig. 4.

Argumentation finding itself in the logic world. A typical argumentation network is the one of Fig. 1. As discussed under (S1) of Sect. 2.2, this network can find itself in the logic world as Exercise 4 mentioned in (S1) of Sect. 2.2. This direction is mathematically complex, as seen from Remark 2.12 (in fact we prove a stronger mathematical theorem for this direction).

We now want to clarify the concept of one system **S1** acting as a metalevel language to describe another system **S2**. We do this by example, and once the reader understands what we mean, we can compare our paper with the papers [5,6,15,17,28,38].

Our starting point is classical predicate logic with term symbols (constants). We want to describe the Hilbert system of intuitionistic implication with axioms (1) and (2), modus ponens and substitution. To achieve this we must say what is an atom, a formula, an axiom and a theorem.

Let $\{q_1, q_2, \ldots\}$ be the atomic wffs of **I**. We use in predicate logic the constants $\{\mathbf{q}_1, \mathbf{q}_2, \ldots\}$ to represent them. Let the function symbol **Imp** represent $\Longrightarrow$. Let the predicates **Atom** and **Formula** represent the notion of atomic formula and a general wff of **I**. Let the predicate **axiom** represent the notion of an axiom of **I** and the predicate **Theorem** represent the notion of a theorem of **I**. All the above are predicates and functions in predicate logic.

We now describe **I**. This will be a predicate logic theory $\Delta(\mathbf{I})$.

(P1) $\mathbf{Atom}(\mathbf{q}_i), i = 1, 2, \ldots$
(P2) $\forall x(\mathbf{Atom}(x) \rightarrow \mathbf{Formula}(x))$
$\forall xy(\mathbf{Formula}(x) \wedge \mathbf{Formula}(y) \rightarrow \mathbf{Formula}(\mathbf{Imp}(x, y)))$
(P3) $\forall xy(\mathbf{Axiom}(\mathbf{Imp}(x, \mathbf{Imp}(y, x)))$
$\forall xy(\mathbf{Axiom}(\mathbf{Imp}(\mathbf{Imp}(x, \mathbf{Imp}(y, z)), \mathbf{Imp}(\mathbf{Imp}(x, y), \mathbf{Imp}x, z))))$
(P4) $\forall xy(\mathbf{Axiom}(x) \rightarrow \mathbf{Theorem}(x))$
(P5) $\forall xy(\mathbf{Theorem}(x) \wedge \mathbf{Theorem}(\mathbf{Imp}(x, y)) \rightarrow \mathbf{Theorem}(y))$

We now got a theory $\Delta(\mathbf{I})$ of first order logic describing **I**. We can use a resolution theorem prover to ask for example does $\Delta(\mathbf{I})$ prove in classical logic $\mathbf{Theorem}(\mathbf{Imp}(x, x))$? This means does **I** prove $a \Longrightarrow a$?

We use this example to clarify the following concepts.

1. One logic/system **S1** talks about another system **S2** in the metalevel.
2. One system, say **S2** is used by another system **S1**, as a case study application.

   In our detailed example, classical logic was describing **I** as a case study application acting as a metalevel system when describing **I**.

Now that we have these concepts, let us compare with other papers.

Grossi's paper [28] uses modal language as a metalevel langauge describing argumentation.

Paper [17] is a survey paper of argumentation theory in general.

Paper [15] discusses with examples, various ways of looking at argumentation theory. It is well worth reading. It is written in the spirit of the current paper but it was written before we discovered the "equivalence" described in the current paper.

Paper [5] characterises extensions in set-theoretic terms. It does not deal with equivalence or connection with classical logic.

Paper [38] uses adaptive logic as a metalevel language for reasoning about argumentation and extensions.

The book [6] uses classical logic as a case study application of argumentation. It uses wffs of classical logic as arguments and defines various attack relations in terms of classical logic consistency. It is strongly related to logic but has no bearing on the question of the equivalence of logic with $\Downarrow$ and argumentation networks. To make the point crystal clear, note that we can take another case study where the arguments are restaurant menus for first course and main course for dinner. We have a chef who tells us what goes with which dish and this is the attack relation.

In summary, we see that none of the above papers do what we do here and they seem related to us because they have "logic" in the title.

We now proceed to address some possible criticism of our work.

The pure logician may say that the Dagger connective is not central to logic. Much more important are the traditional intuitive connectives **and, or, not**, and **implies**. The Sheffer stroke and the dagger were studied in logic for technical reasons. The logic community was interested in connectives which are functionally complete (the Sheffer stroke and the Peirce–Quine dagger) and can define all other connectives and can be axiomatised by a minimal number of axioms containing a minimal number of letters. However, such connectives are not so important. So if the argumentation attack relation is essentially the Dagger connective, then good for argumentation. We the logicians are not necessarily interested.

My answer to this is to say that we are generalising the concept of what is a logical system, a foundational issue for logic, and advise the logician to take a look at Sect. 3. This might help.

To the argumentation people I would say that first they should realise that they have the power of classical logic in a natural way. The correspondence shown in Table 2 and discussed in (S1) and (S2) in Sect. 2.2 and further methodologically described in Sect. 6 above is very natural. So they should not be surprised at what one can do with argumentation. See, for example, Sect. 2.4.

The argumentation researcher may come up with three objections:

1. The correspondence in Table 2 is for a very restricted argumentation frame, namely The canonical argumentation frame based on Dagger logic of Definition 2.6 which is basically a tree with small cycles at the top as in Fig. 4. This is a very restricted argumentation frame and all the richness of argumentation is lost.

   To this we answer that this canonical frame is only our starting point, showing how logic can find itself in argumentation. We get the richness of all argumentation frames with cycles, by applying equivalence relations to this starting frame and obtaining new factor frames. This is done in Sect. 2.4 and also see Remark 4.8 where higher level

attacks is simplified and is reduced to logic. Section 4 gives as predicate argumentation suggested by the correspondence with logic.

2. The second objection is more tricky to answer. It claims that ordinary argumentation is simple and that the connection with logic makes things more complicated. So it can't be useful. My answer is that it is not more complicated, just a new natural way of looking at argumentation. The correspondence is natural and anyone used to logic can see it. Real complications arise when the new representations are completely different paradigm, say something like category theory. Logic is already related to and is firmly embedded in argumentation!
3. The third objection is linguistic. We talk about classical logic and yet we end up needing three valued logic, {in, out, undecided}. My answer to this is two fold.

   First, having functions undefined on some elements of its domain, was never considered a departure from classical logic. Look at the field of partial recursive functions, it is one of the pillars of classical logic!

   We talk about three values for convenience and also in anticipation of paper [24], where we use continuous values in [0,1]. But the equational model of [24] is a different interpretation altogether and equally applies to Logic or to Argumentation.

   Second, from the semantic point of view it can be argued, that any non-classical logic which can be characterised by a finite matrix is essentially classical logic. This is because the matrix can be expressed in classical logic. Of course this is a semantical point of view, which includes the Semantic Tableaux formulation as well. However, when it comes to writing a Hilbert system or a Gentzen system for such logics it can be extremely difficult and much different from classical logic. See our book [31].

Further, I stress that Argumentation needs a wider theoretical basis, not just for methodological reasons but for the social cohesiveness of the community. Without a good logical connection and meta-logical foundations, there is the danger that the community will fragment and be absorbed into the variety of its application areas. Those who apply argumentation to Law will get absorbed in the law community, those who apply argumentation to Agent Theory will be absorbed in the agents community. History shows us that this is how the Logic Programming community fragmented! In fact the Logic Programming community was lucky; because of its close connection with logic, a vibrant core group did survive. I am not sure this would happen to the argumentation community. The strength of argumentation is in its applicability across many diverse disciplines (much more than Logic Programming) and that can be a problem!

Now I have something to say to both communities.

It is clear from what we are doing that the most general formal setup is that of a network (S, R) with various annotation to the elements of the network and some algorithms, relying on the geometry of the network and

the annotation, for going around the network updating the annotations, possibly in loops, possibly never terminating seeking steady state solutions to the annotation function.

This is the most general case and it integrates logic, argumentation networks, neural networks, ecology networks and more.

We have done this for numerical values in our papers [1,2,24].

The important point here is that the general integrating framework is natural in itself. It has a meaning in itself and logic and networks each can be identified (I have not written this yet) internally as specialised case of the general framework. This is an important test showing the integration is natural!

It is not the case that in this integration each component loses its identity. Not at all, they give each other ideas, being residents of a more general framework.

We saw in Sects. 2 and 3 that logic can be viewed as a network. However, Logic is also strong enough to describe and talk about networks. There is a feature logic can do which is not available in networks—and this is the ease of the interaction of object level and meta-level features. Can we expand networks to be able to talk about itself?

This problem we still have to solve, namely how to embed meta-level features of networks in the object level of networks themselves. In other words how can networks talk about networks much in the same way that logic can talk about logic.

We did start to address this in the predicate logic section, Sect. 4. Roughly we showed a correspondence between meta-predicates in logic and attacks on one network from another network. So for a network to talk about itself it must have certain attacks from one part of itself to another part, but we leave this for future papers, or maybe include it in our planned book on argumentation [25]!

We close this section by listing what future papers we can write on topics hinted in this paper.

1. The ideas in Sect. 4 can give rise to several papers on predicate argumentation and a systematic study of various papers in the literature which can be simplified and integrated using predicate argumentation.
2. The ideas of Sect. 5, where we replace classical propositional logic by linear logic, naturally give us a handle on resource considerations in argumentation. We do not need to define resource features ad hoc but we can be systematic. Use arguments in attack only once, or if you keep using the same argument again and again, it weakens because people get weary of it, and more.
3. The results of Sect. 3 can be further developed in a systematic way into a paper: "What is a logical system, version 2011".

Acknowledgements

I am grateful to Alexander Bochman, Philippe Besnard, Martin Caminada, and Leon van der Torre for valuable comments. I am also grateful to the four

referees of *Logica Universalis* for their detailed comments and penetrating criticism.

References

[1] Barringer, H., Gabbay, D., Woods, J.: Temporal dynamics of argumentation networks. In: Hutter, D., Stephan, W. (eds.) Volume Dedicated to Joerg Siekmann. Mechanising Mathematical Reasoning. LNCS, vol. 2605, pp. 59–98. Springer, Berlin (2005)

[2] Barringer, H., Gabbay, D., Woods, J.: Network modalities. In: Gross, G., Schulz, K.U. (eds.) Linguistics, Computer Science and Language Processing. Festschrift for Franz Guenthner on the Occasion of his 60th Birthday, pp. 79–102. College Publications (2008)

[3] Bench-Capon, T.J.M.: Persuasion in practical argument using value-based argumentation frameworks. J. Logic Comput. **13**(3), 429–448 (2003)

[4] Bench-Capon, T.J.M., Atkinson, K.: Abstract argumentation and values. In: Rahwan, I., Simari, G. (eds.) Argumentation in AI, Chap. 3, pp. 45–64. Springer, Berlin (2009)

[5] Besnard, P., Doutre, S.: Characterization of semantics for argument systems. In: Dubois, D., Welty, C., Williams, OM.-A (eds.) KR 2004, pp. 183–193. AAAI Press (2004)

[6] Besnard, P., Hunter, A.: Elements of Argumentation. MIT Press (2008)

[7] Boella, G., Gabbay, D., van der Torre, L., Villata, S.: Support in Abstract Argumentation (n383) Expanded Version. This paper is an expansion of an earlier original paper under the same title published in 2010: In: Baroni, P., Cerutti, F., Giacomi, M., Simari, G. (eds.) Computational Models of Argument, COMMA 2010, pp. 111–122. IOS press (2010)

[8] Baroni, P, Cerutti, F., Giacomin, M., Guida, G.: Encompassing attacks to attacks in abstract argumentation frameworks. In: ECSQARU 2009, LNAI, vol. 5590, pp. 83–94. Springer, Berlin (2009)

[9] Brewka, G., Woltran, S.: Abstract dialectical frameworks. In: Proceedings of KR 2010. AAAI Press (2010)

[10] Brewka, G., Dunne, P., Woltran, S.: Relating the semantics of abstract dialectical frameworks and standard AFs. In: Proceedings of IJCAI-11, pp. 780–786 (2011)

[11] Caminada, M.: On the issue of reinstatement in argumentation. In: Fischer, M., van der Hoek, W., Konev, B., Lisitsa, A. (eds.) Logics in Artificial Intelligence: 10th European Conference, JELIA 2006. LNAI, vol. 4160, pp. 111–123. Springer, Berlin (2006)

[12] Caminada, M.: Semi-stable semantics. In: Dunne, P.E., Bench-Capon, T.J.M. (eds.) Computational Models of Argument. Proceedings of COMMA 2006, pp. 121–130. IOS Press (2006)

[13] Caminada, M., Wu, Y.: On the limitations of abstract argumentation. In: Proceedings of BNAIC 2011. http://users.numericable.lu/martincaminada/publications/BNAIC_limitations_abstract.pdf

[14] Caminada, M., Amgoud, L.: On the evaluation of argumentation formalisms. Artif. Intell. **171**(5–6), 286–310 (2007)

[15] Caminada, M., Gabbay, D.: A logical account of formal argumentation Studia Logica **93**, 109–145 (2009)

[16] Chalamish, M., Gabbay, D., Schild, U.: Intelligent evaluation of evidence using Wigmore diagrams. In: ICAIL-11, pp. 61–65 (2011)

[17] Chesnevar, C., Maguitman, A., Loui, R.: Logical models of argument. ACM Comput. Surv. **32**, 337–383 (2000)

[18] Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell. **77**, 321–357 (1995)

[19] Gabbay, D.: Labelled Deductive Systems. Oxford University Press (1996)

[20] Gabbay, D.: Fibring Logics. Oxford University Press (1998)

[21] Gabbay, D.: Modal provability foundations for argumentation networks. Studia Logica **93**(2–3), 181–189 (2009)

[22] Gabbay, D.: Fibering argumentation networks. Studia Logica **93**(2–3), 231–296 (2009)

[23] Gabbay, D.: Semantics for higher level attacks in extended argumentation fames. Part 1: overview. Studia Logica **39**, 355–379 (2009)

[24] Gabbay, D.: Equational Approach to Argumentation Networks (2011)

[25] Gabbay, D.: Meta-Logical Investigations in Argumentation Networks. Monograph. Springer

[26] Gabbay, D., d'Avila Garcez, A.S.: Logical modes of attack in argumentation networks. Studia Logica **93**(2–3), 199–230 (2009)

[27] Gamut, L.T.F.: Logic, Language, and Meaning. University of Chicago Press (1991)

[28] Grossi, D.: On the logic of abstract argumentation. In: AAMAS'10 (2010)

[29] Hunter, A.: Base logics in argumentation. In: Proceedings of COMMA 2010, pp. 275–286 (2010)

[30] Leisenring, A.C.: Mathematical logic and Hilbert's epsilon-symbol. McDonald, London (1969)

[31] Metcalfe, G., Olivetti, N., Gabbay, D.: Proof Theory for Fuzzy Logics. Springer (2008)

[32] Modgil, S.: Reasoning about preferences in argumentation frameworks. Artif. Intell. **173**(9–10), 901–993 (2009)

[33] Peirce, C.S.: A Boolean algebra with one constant. In: Hartshorne, C., Weiss, P. (eds.) Collected Papers of Charles Sanders Peirce, vol. 4, pp. 12–20. Harvard University Press (1931–1935)

[34] Prakken, H.: An abstract framework for argumentation with structured arguments. Argum. Comput. **1**(2), 93–124 (2010)

[35] Price, R.: The Stroke function in natural deduction. Zeitsehr. f.math. Logik und Grundlagen d. Math. **7**, 117–128 (1961)

[36] Seto, Y.: Proofs of some axioms by stroke function. In: Proc. Japan Acad. vol. 44, pp. 1024–1026 (1968)

[37] Sheffer, H.M.: A set of five independent postulates for Boolean algebras, with application to logical constants. Trans. Am. Math. Soc. **14**, 481–488 (1913)

[38] Strasser, S.: Towards the proof-theoretic unification of Dung's argumentation framework: an adaptive logic approach. J. Logic Comput. **21**, 133–156 (2011)

Dov M. Gabbay
Bar Ilan University
Ramat-Gan, Israel
and
King's College London
London, UK
e-mail: dov.gabbay@kcl.ac.uk
and
University of Luxembourg
Luxembourg City
Luxembourg

Received: June 20, 2011.
Accepted: August 20, 2011.