Argumentation among Agents

Iyad Rahwan

[1]Masdar Institute of Science & Technology, UAE
[2]University of Edinburgh, UK
[3]Massachusetts Institute of Technology, USA

I. Rahwan. *Argumentation among Agents*. In: G. Weiss (Ed.), Multiagent Systems, 2nd edition, MIT Press, 2012.

What is Argumentation?

What philosophers call it!

Arguing with Others

"A verbal and social activity of reason aimed at increasing (or decreasing) the acceptability of a controversial standpoint for the listener or reader, by putting forward a constellation of propositions (i.e. arguments) intended to justify (or refute) the standpoint before a rational judge" [van Eemeren et al]

"the giving of reasons to support or criticize a claim that is questionable, or open to doubt" [Walton]

Argumentation versus Reasoning

If you are the judge, argumentation becomes (nonmonotonic) reasoning.

Process of Argumentation

1. Constructing *arguments* (in *favor* of / *against* a “statement”) from available information.

   A: “Tweety is a bird, so it flies”
   B: “Tweety is just a cartoon!”

2. Determining the different *conflicts* among the arguments.

   “Since Tweety is a cartoon, it cannot fly!” (B attacks A)

3. Evaluating the *acceptability* of the different arguments.

   “Since we have no reason to believe otherwise, we’ll assume Tweety is a cartoon.” (accept B). “But then, this means despite being a bird he cannot fly.” (reject A).

4. Concluding, or defining the *justified conclusions*.

   “We conlcude that Tweety cannot fly!”

Process of Argumentation

1. Constructing *arguments* (in *favor* of / *against* a "statement") from available information.

   A: "Tweety is a bird, so it flies"
   B: "Tweety is just a cartoon!"

2. Determining the different *conflicts* among the arguments.

   "Since Tweety is a cartoon, it cannot fly!" (B attacks A)

3. Evaluating the *acceptability* of the different arguments.

   "Since we have no reason to believe otherwise, we'll assume Tweety is a cartoon." (accept B). "But then, this means despite being a bird he cannot fly." (reject A).

4. Concluding, or defining the *justified conclusions*.

   "We conlcude that Tweety cannot fly!"

Process of Argumentation

1. Constructing *arguments* (in *favor* of / *against* a “statement”) from available information.

   A: “Tweety is a bird, so it flies”
   B: “Tweety is just a cartoon!”

2. Determining the different *conflicts* among the arguments.

   “Since Tweety is a cartoon, it cannot fly!” (B attacks A)

3. Evaluating the *acceptability* of the different arguments.

   “Since we have no reason to believe otherwise, we’ll assume Tweety is a cartoon.” (accept B). “But then, this means despite being a bird he cannot fly.” (reject A).

4. Concluding, or defining the *justified conclusions*.

   “We conlcude that Tweety cannot fly!”

Process of Argumentation

1. Constructing *arguments* (in *favor* of / *against* a “statement”) from available information.

   A: “Tweety is a bird, so it flies”
   B: “Tweety is just a cartoon!”

2. Determining the different *conflicts* among the arguments.

   “Since Tweety is a cartoon, it cannot fly!” (B attacks A)

3. Evaluating the *acceptability* of the different arguments.

   “Since we have no reason to believe otherwise, we’ll assume Tweety is a cartoon.” (accept B). “But then, this means despite being a bird he cannot fly.” (reject A).

4. Concluding, or defining the *justified conclusions*.

   “We conlcude that Tweety cannot fly!”

Outline

Outline

Outline

Argumentation System

Example model by Prakken

A tuple $(\mathcal{L}, -, \mathcal{R}_s, \mathcal{R}_d, \leq)$

- a logical language $\mathcal{L}$,
- strict rules $\mathcal{R}_s$
- defeasible rules $\mathcal{R}_d$
- partial order $\leq$ over $\mathcal{R}_d$
- *Contrariness function* $- : \mathcal{L} \rightarrow 2^{\mathcal{L}}$ captures conflict between formulas

  Classical negation $\neg$ captured by $\neg\varphi \in \overline{\varphi}$ and $\varphi \in \overline{\neg\varphi}$.

Knowledge Base

A particular knowledge base $(\mathcal{K}, \leq')$ with:

- $\mathcal{K} \subseteq \mathcal{L}$ divided into:
  - $\mathcal{K}_n$ are necessary *axioms* (cannot be attacked);
  - $\mathcal{K}_p$ are *ordinary premises*;
  - $\mathcal{K}_a$ are *assumptions*;
  - $\mathcal{K}_i$ are *issues*.
- $\leq'$ is a partial order on $\mathcal{K} \backslash \mathcal{K}_n$.

Inference rules:

- *defeasible* rule $\varphi_1, \ldots, \varphi_n \Rightarrow \varphi$ means conclusion φ follows *presumably* from the premises $\varphi_1, \ldots, \varphi_n$
- *strict* rule $\varphi_1, \ldots, \varphi_n \rightarrow \varphi$ stands for classical implication

Functions *Perm*(A), *Conc*(A) and *Sub*(A) returns premises, conclusion, and *sub-arguments* of argument A respectively.

Knowledge Base

A particular knowledge base $(\mathcal{K}, \leq')$ with:

- $\mathcal{K} \subseteq \mathcal{L}$ divided into:
  - $\mathcal{K}_n$ are necessary *axioms* (cannot be attacked);
  - $\mathcal{K}_p$ are *ordinary premises*;
  - $\mathcal{K}_a$ are *assumptions*;
  - $\mathcal{K}_i$ are *issues*.
- $\leq'$ is a partial order on $\mathcal{K} \backslash \mathcal{K}_n$.

Inference rules:

- *defeasible* rule $\varphi_1, \ldots, \varphi_n \Rightarrow \varphi$ means conclusion φ follows *presumably* from the premises $\varphi_1, \ldots, \varphi_n$
- *strict* rule $\varphi_1, \ldots, \varphi_n \rightarrow \varphi$ stands for classical implication

Functions *Perm*(A), *Conc*(A) and *Sub*(A) returns premises, conclusion, and *sub-arguments* of argument A respectively.

Argument

An argument is any of the following:

- $\varphi \in \mathcal{K}$, where *Prem*$(A) = \{\varphi\}$, *Conc*$(A) = \varphi$, and *Sub*$(A) = \{\varphi\}$.
- $A_1, \ldots, A_n \rightarrow \psi$, where $A_1, \ldots, A_n$ are arguments, and there exists in $\mathcal{R}_s$ a strict rule $Conc(A_1), \ldots, Conc(A_n) \rightarrow \psi$.
- $A_1, \ldots, A_n \Rightarrow \psi$, where $A_1, \ldots, A_n$ are arguments, and there exists in $\mathcal{R}_d$ a defeasible rule $Conc(A_1), \ldots, Conc(A_n) \rightarrow \psi$.

where

- $Prem(A) = Prem(A_1) \cup \cdots \cup Prem(A_n)$
- $Sub(A) = Sub(A_1) \cup \cdots \cup Sub(A_n) \cup \{A\}$

Example

Due to Prakken

With this knowledge base:

- $\mathcal{R}_s = \{p, q \rightarrow s; u, v \rightarrow w\}$, $\mathcal{R}_d = \{p \Rightarrow t; s, r, t \Rightarrow v\}$,
- $\mathcal{K}_n = \{q\}$, $\mathcal{K}_p = \{p, u\}$, $\mathcal{K}_a = \{r\}$.

We can construct the following arguments:

$A_1 : p$ $A_3 : r$ $A_5 : A_1 \Rightarrow t$ $A_7 : A_3, A_5, A_6 \Rightarrow v$

$A_2 : q$ $A_4 : u$ $A_6 : A_1, A_2 \rightarrow s$ $A_8 : A_4, A_7 \rightarrow w$

With:

- $Prem(A_8) = \{p, q, r, u\}$, $Conc(A_8) = \{w\}$,
- $Sub(A_8) = \{A_1, A_2, A_3, A_4, A_5, A_6, A_6, A_8\}$.

Example

Due to Prakken

With this knowledge base:

- $\mathcal{R}_s = \{p, q \rightarrow s; u, v \rightarrow w\}$, $\mathcal{R}_d = \{p \Rightarrow t; s, r, t \Rightarrow v\}$,
- $\mathcal{K}_n = \{q\}$, $\mathcal{K}_p = \{p, u\}$, $\mathcal{K}_a = \{r\}$.

We can construct the following arguments:

$A_1 : p$	$A_3 : r$	$A_5 : A_1 \Rightarrow t$	$A_7 : A_3, A_5, A_6 \Rightarrow v$
$A_2 : q$	$A_4 : u$	$A_6 : A_1, A_2 \rightarrow s$	$A_8 : A_4, A_7 \rightarrow w$

With:

- $Prem(A_8) = \{p, q, r, u\}$, $Conc(A_8) = \{w\}$,
- $Sub(A_8) = \{A_1, A_2, A_3, A_4, A_5, A_6, A_6, A_8\}$.

Example

Due to Prakken

With this knowledge base:

- $\mathcal{R}_s = \{p, q \to s; u, v \to w\}$, $\mathcal{R}_d = \{p \Rightarrow t; s, r, t \Rightarrow v\}$,
- $\mathcal{K}_n = \{q\}$, $\mathcal{K}_p = \{p, u\}$, $\mathcal{K}_a = \{r\}$.

We can construct the following arguments:

$A_1 : p$	$A_3 : r$	$A_5 : A_1 \Rightarrow t$	$A_7 : A_3, A_5, A_6 \Rightarrow v$
$A_2 : q$	$A_4 : u$	$A_6 : A_1, A_2 \to s$	$A_8 : A_4, A_7 \to w$

With:

- $Prem(A_8) = \{p, q, r, u\}$, $Conc(A_8) = \{w\}$,
- $Sub(A_8) = \{A_1, A_2, A_3, A_4, A_5, A_6, A_6, A_8\}$.

Attack Among Arguments

- $\mathcal{R}_s = \{p, q \rightarrow s; u, v \rightarrow w\}$, $\mathcal{R}_d = \{p \Rightarrow t; s, r, t \Rightarrow v\}$,
- $\mathcal{K}_n = \{q\}$, $\mathcal{K}_p = \{p, u\}$, $\mathcal{K}_a = \{r\}$.

Arguments:

$A_1 : p$	$A_3 : r$	$A_5 : A_1 \Rightarrow t$	$A_7 : A_3, A_5, A_6 \Rightarrow v$
$A_2 : q$	$A_4 : u$	$A_6 : A_1, A_2 \rightarrow s$	$A_8 : A_4, A_7 \rightarrow w$

With:

- $Prem(A_8) = \{p, q, r, u\}$, $Conc(A_8) = \{w\}$,
- $Sub(A_8) = \{A_1, A_2, A_3, A_4, A_5, A_6, A_6, A_8\}$.

Undercut

An argument can *undercut* another argument by showing that a defeasible rule cannot be applied.

A_8 can be undercut by an argument with conclusion $\overline{A_5}$, since argument A_5 is constructed with a defeasible rule.

Attack Among Arguments

- $\mathcal{R}_s = \{p, q \rightarrow s; u, v \rightarrow w\}$, $\mathcal{R}_d = \{p \Rightarrow t; s, r, t \Rightarrow v\}$,
- $\mathcal{K}_n = \{q\}$, $\mathcal{K}_p = \{p, u\}$, $\mathcal{K}_a = \{r\}$.

Arguments:

$A_1 : p$	$A_3 : r$	$A_5 : A_1 \Rightarrow t$	$A_7 : A_3, A_5, A_6 \Rightarrow v$
$A_2 : q$	$A_4 : u$	$A_6 : A_1, A_2 \rightarrow s$	$A_8 : A_4, A_7 \rightarrow w$

With:

- $Prem(A_8) = \{p, q, r, u\}$, $Conc(A_8) = \{w\}$,
- $Sub(A_8) = \{A_1, A_2, A_3, A_4, A_5, A_6, A_6, A_8\}$.

Rebut

An argument can *rebut* another argument by supporting the opposite conclusion.

A_8 can be rebutted on A_5 with an argument with conclusion $\overline{t}$, or rebutted on A_7 with an argument with conclusion $\overline{v}$.

Attack Among Arguments

- $\mathcal{R}_s = \{p, q \to s; u, v \to w\}$, $\mathcal{R}_d = \{p \Rightarrow t; s, r, t \Rightarrow v\}$,
- $\mathcal{K}_n = \{q\}$, $\mathcal{K}_p = \{p, u\}$, $\mathcal{K}_a = \{r\}$.

Arguments:

$A_1 : p$	$A_3 : r$	$A_5 : A_1 \Rightarrow t$	$A_7 : A_3, A_5, A_6 \Rightarrow v$
$A_2 : q$	$A_4 : u$	$A_6 : A_1, A_2 \to s$	$A_8 : A_4, A_7 \to w$

With:

- $Prem(A_8) = \{p, q, r, u\}$, $Conc(A_8) = \{w\}$,
- $Sub(A_8) = \{A_1, A_2, A_3, A_4, A_5, A_6, A_6, A_8\}$.

Undermine (or Premise Attack)

An argument can *undermine* another argument by attacking one of its premises.

A_8 can be undermined by an argument with conclusion $\overline{p}$, $\overline{r}$, or $\overline{u}$.

Defeat among Arguments

Defeat

Argument A *defeats* argument B if the former attacks the latter, and is also preferred to it according to some preference relation $\prec$.

Relation $\prec$ may itself be based on the nature of the attack as well as the preference relations $\leq$ and $\leq'$ over the formulas involved.

Outline

Argumentation Scheme

Argumentation schemes are forms (or categories) of argument, representing stereotypical ways of drawing inferences from particular patterns of premises to conclusions in a particular domain (e.g. reasoning about action).

For each scheme, we list:

1. Premises
2. Conclusion
3. A set of *critical questions* that can be used to scrutinize the argument by questioning explicit or implicit premises.

Various formal and semi-formal models have been proposed.

Argumentation Scheme Example

Walton's "sufficient condition scheme for practical reasoning":

In the current circumstances R
We should perform action A
Which will result in new circumstances S
Which will realise goal G
Which will promote some value V.

Associated critical questions include:

CQ1: *Are the believed circumstances true?*
CQ2: *Does the action have the stated consequences?*
CQ3: *Assuming the circumstances and that the action has the stated consequences, will the action bring about the desired goal?*
CQ4: *Does the goal realise the value stated?*
CQ5: *Are there alternative ways of realising the same consequences?*

Argumentation Scheme Example

Walton's "sufficient condition scheme for practical reasoning":

In the current circumstances R
We should perform action A
Which will result in new circumstances S
Which will realise goal G
Which will promote some value V.

Associated critical questions include:

CQ1: *Are the believed circumstances true?*
CQ2: *Does the action have the stated consequences?*
CQ3: *Assuming the circumstances and that the action has the stated consequences, will the action bring about the desired goal?*
CQ4: *Does the goal realise the value stated?*
CQ5: *Are there alternative ways of realising the same consequences?*

Outline

General Idea

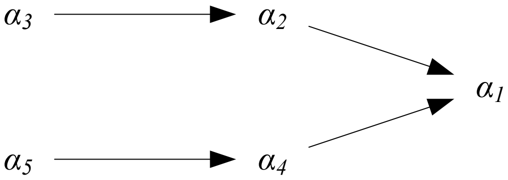
- Ignore internal structure of arguments
- An argument is just a node
- Focus only on the defeat structure

Argumentation Framework

An *argumentation framework* is a pair $AF = \langle \mathcal{A}, \rightharpoonup \rangle$ where $\mathcal{A}$ is a finite set of arguments and $\rightharpoonup \subseteq \mathcal{A} \times \mathcal{A}$ is a defeat relation. We say that an argument α *defeats* an argument β if $(\alpha, \beta) \in \rightharpoonup$ (sometimes written $\alpha \rightharpoonup \beta$).

Argument Graphs: Example

Argument α_1 has two defeaters (i.e. counter-arguments) α_2 and α_4, which are themselves defeated by arguments α_3 and α_5 respectively.



We will focus on these structures from now on.

Despite their simplicity, they are very powerful.

Outline

Collectively Evaluating Arguments

How do we evaluate arguments based on the defeat structure?

Let $S^+ = \{\beta \in \mathcal{A} \mid \alpha \rightharpoonup \beta \text{ for some } \alpha \in S\}$.
Let $\alpha^- = \{\beta \in \mathcal{A} \mid \beta \rightharpoonup \alpha\}$.

Conflict-free, Defense

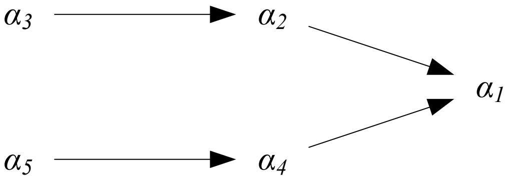
Let $\langle \mathcal{A}, \rightharpoonup \rangle$ be an argumentation framework and let $S \subseteq \mathcal{A}$ and let $\alpha \in \mathcal{A}$.

- S is *conflict-free* if $S \cap S^+ = \emptyset$.
- S *defends* argument α if $\alpha^- \subseteq S^+$. We also say that argument α is *acceptable with respect to S*.

Make Titles Informative.

Intuitively:

- A set of arguments is *conflict free* if no argument in that set defeats another.
- A set of arguments *defends* a given argument if it defeats all its defeaters.



In the above graph, $\{\alpha_3, \alpha_5\}$ defends α_1.

Charactarizing Defense

Characteristic Function

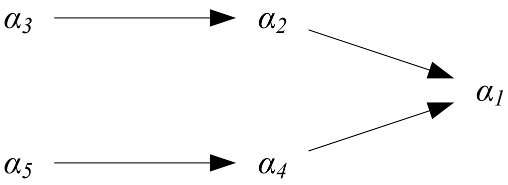
The *characteristic function* of an argumentation framework is $\mathcal{F}$:$2^{\mathcal{A}} \rightarrow 2^{\mathcal{A}}$ such that, given $S \subseteq \mathcal{A}$, we have $\mathcal{F}(S) = \{\alpha \in \mathcal{A} \mid S \text{ defends } \alpha\}$.

Admissibility

Let S be a conflict-free set of arguments in framework $\langle \mathcal{A}, \rightharpoonup \rangle$. S is *admissible* if it is conflict-free and defends every element in S (i.e. if $S \subseteq \mathcal{F}(S)$).

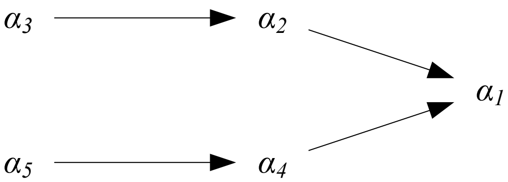
Intuitively, a set of arguments is *admissible* if it is a conflict-free set that defends itself against any defeater

Admissibility Example



- Sets $\emptyset$, $\{\alpha_3\}$, $\{\alpha_5\}$, and $\{\alpha_3, \alpha_5\}$ are all admissible simply because they do not have any defeaters.
- $\{\alpha_1, \alpha_3, \alpha_5\}$ is also admissible since it defends itself against defeaters α_2 and α_4.

Admissibility Example



- Sets $\emptyset$, $\{\alpha_3\}$, $\{\alpha_5\}$, and $\{\alpha_3, \alpha_5\}$ are all admissible simply because they do not have any defeaters.
- $\{\alpha_1, \alpha_3, \alpha_5\}$ is also admissible since it defends itself against defeaters α_2 and α_4.

Complete Extension

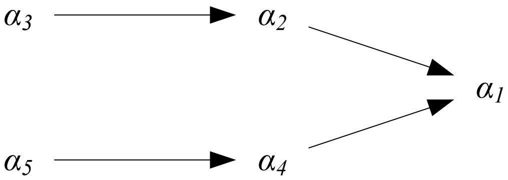
An admissible set S is a *complete extension* if and only if *all* arguments defended by S are also in S.

Complete Extension

Let S be a conflict-free set of arguments in framework $\langle \mathcal{A}, \rightharpoonup \rangle$. S is a *complete extension* if $S = \mathcal{F}(S)$.

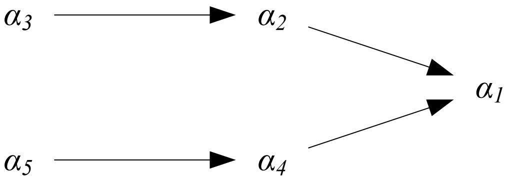
That is, if S is a fixed point of the operator $\mathcal{F}$.

Complete Extension Example



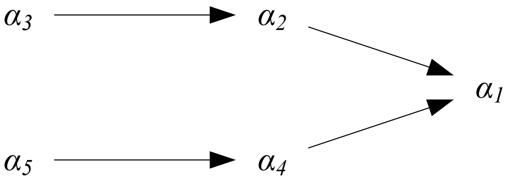
- Admissible set $\{\alpha_3, \alpha_5\}$ is not a complete extension, since it defends α_1 but does not include α_1.
- Sets $\{\alpha_3\}$, $\{\alpha_5\}$ are not complete extensions, since $\mathcal{F}(\{\alpha_3\}) = \{\alpha_3, \alpha_5\}$ and $\mathcal{F}(\{\alpha_5\}) = \{\alpha_3, \alpha_5\}$.
- Admissible set $\{\alpha_1, \alpha_3, \alpha_5\}$ is the only complete extension, since $\mathcal{F}(\{\alpha_1, \alpha_3, \alpha_5\}) = \{\alpha_1, \alpha_3, \alpha_5\}$.

Complete Extension Example



- Admissible set $\{\alpha_3, \alpha_5\}$ is not a complete extension, since it defends α_1 but does not include α_1.
- Sets $\{\alpha_3\}$, $\{\alpha_5\}$ are not complete extensions, since $\mathcal{F}(\{\alpha_3\}) = \{\alpha_3, \alpha_5\}$ and $\mathcal{F}(\{\alpha_5\}) = \{\alpha_3, \alpha_5\}$.
- Admissible set $\{\alpha_1, \alpha_3, \alpha_5\}$ is the only complete extension, since $\mathcal{F}(\{\alpha_1, \alpha_3, \alpha_5\}) = \{\alpha_1, \alpha_3, \alpha_5\}$.

Complete Extension Example



- Admissible set $\{\alpha_3, \alpha_5\}$ is not a complete extension, since it defends α_1 but does not include α_1.
- Sets $\{\alpha_3\}$, $\{\alpha_5\}$ are not complete extensions, since $\mathcal{F}(\{\alpha_3\}) = \{\alpha_3, \alpha_5\}$ and $\mathcal{F}(\{\alpha_5\}) = \{\alpha_3, \alpha_5\}$.
- Admissible set $\{\alpha_1, \alpha_3, \alpha_5\}$ is the only complete extension, since $\mathcal{F}(\{\alpha_1, \alpha_3, \alpha_5\}) = \{\alpha_1, \alpha_3, \alpha_5\}$.

Argument Labeling

A labelling specifies which arguments are:

- Accepted (labelled `in`)
- Rejected (labelled `out`)
- Undecided (labelled `undec`)

Labellings must satisfy the condition:

- An argument is `in` if and only if all of its defeaters are `out`.
- An argument is `out` if and only if at least one of its defeaters is `in`.
- Otherwise, it is undecided.

For any labelling satisfying the conditions above, those arguments that happen to be labelled `in` form a complete extension.

Argument Labeling

A labelling specifies which arguments are:

- Accepted (labelled `in`)
- Rejected (labelled `out`)
- Undecided (labelled `undec`)

Labellings must satisfy the condition:

- An argument is `in` if and only if all of its defeaters are `out`.
- An argument is `out` if and only if at least one of its defeaters is `in`.
- Otherwise, it is undecided.

For any labelling satisfying the conditions above, those arguments that happen to be labelled `in` form a complete extension.

Argument Labeling

A labelling specifies which arguments are:

- Accepted (labelled `in`)
- Rejected (labelled `out`)
- Undecided (labelled `undec`)

Labellings must satisfy the condition:

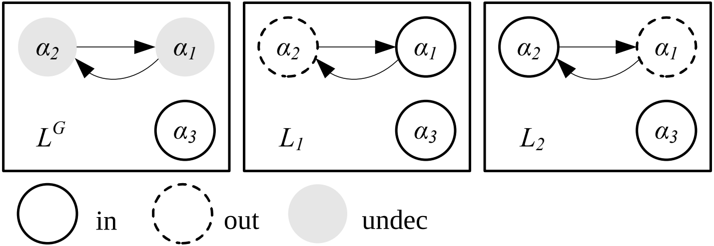
- An argument is `in` if and only if all of its defeaters are `out`.
- An argument is `out` if and only if at least one of its defeaters is `in`.
- Otherwise, it is undecided.

For any labelling satisfying the conditions above, those arguments that happen to be labelled `in` form a complete extension.

Labeling Examples

Graph with Three Legal Labelings

- An argument is `in` if and only if all of its defeaters are `out`.
- An argument is `out` if and only if at least one of its defeaters is `in`.
- Otherwise, it is undecided.



Three complete extensions: $\{\alpha_3\}$, $\{\alpha_1, \alpha_3\}$ and $\{\alpha_2, \alpha_3\}$.

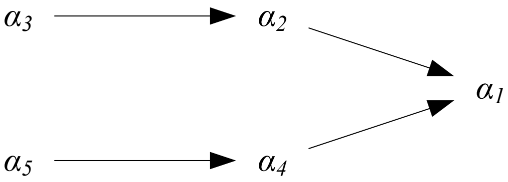
Refinements of the Complete Extension

Refinements of Complete Semantics

Let S be a conflict-free set of arguments in framework $\langle \mathcal{A}, \rightharpoonup \rangle$.

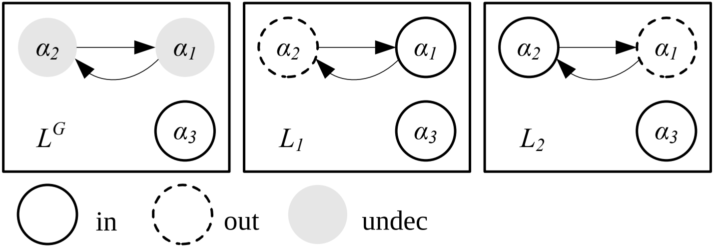
- S is a *grounded extension* if it is the minimal (w.r.t. set-inclusion) complete extension (or, alternatively, if S is the least fixed-point of $\mathcal{F}(.)$).
- S is a *preferred extension* if it is a maximal (w.r.t. set-inclusion) complete extension (or, alternatively, if S is a maximal admissible set).
- S is a *stable extension* if $S^+ = \mathcal{A} \backslash S$.
- S is a *semi-stable extension* if S is a complete extension of which $S \cup S^+$ is maximal.

Extensions: Examples



$\{\alpha_1, \alpha_3, \alpha_5\}$ is the only preferred extension.

Extensions: Examples



Two preferred extensions, which are the maximal complete extension w.r.t. set inclusion.

- $\{\alpha_1, \alpha_3\}$
- $\{\alpha_2, \alpha_3\}$

And one grounded extension: $\{\alpha_3\}$

Extension vs. Labeling

Let:

$\texttt{in}(L) = \{\alpha \in \mathcal{A} \mid L(\alpha) = \texttt{in}\}$
$\texttt{out}(L) = \{\alpha \in \mathcal{A} \mid L(\alpha) = \texttt{out}\}$
$\texttt{undec}(L) = \{\alpha \in \mathcal{A} \mid L(\alpha) = \texttt{undec}\}$

Extensions	**Restrictions on Labellings**
complete	all labellings
grounded	minimal `in`, or equivalently minimal `out`, or equivalently maximal `undec`
preferred	maximal `in`, or equivalently maximal `out`
semi-stable	minimal `undec`
stable	empty `undec`

Argument Status

Argument Status

Let $\langle \mathcal{A}, \rightharpoonup \rangle$ be an argumentation system, and $\mathcal{E}_1, \ldots, \mathcal{E}_n$ its extensions under a given semantics. Let $\alpha \in \mathcal{A}$.

1. α is *sceptically accepted* iff $\alpha \in \mathcal{E}_i, \forall \mathcal{E}_i$ with $i = 1, \ldots, n$.
2. α is *credulously accepted* iff $\exists \mathcal{E}_i$ such that $\alpha \in \mathcal{E}_i$.
3. α is *rejected* iff $\nexists \mathcal{E}_i$ such that $\alpha \in \mathcal{E}_i$.

Intuitively, an argument is:

- *sceptically accepted* if it can be accepted without making any hypotheses beyond what is surely self-defending).
- *credulously accepted* if there is a possible consistent set of hypotheses in which it is consistent.
- otherwise, there is no basis for accepting it.

Outline

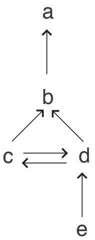
Outline

1. What is an Argument?
   - Arguments as Chained Inference Rules
   - Arguments as Instances of Schemes
   - Arguments as Graphs
2. Evaluating Arguments
3. Argumentation Protocols
   - Abstract Argument Games
   - Dialogue Systems
4. Strategic Argumentation and Game Theory
   - Game Theory Background
   - Argumentation Mechanism Design
   - Case Study: Grounded Semantics
5. Argument Interchange Format

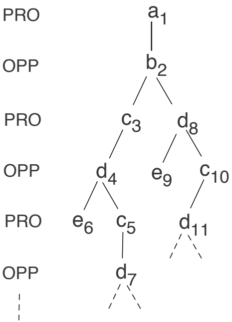
What is an Abstract Argument Game?

- Typically two agents:
  - PRO (the proponent)
  - OPP (the opponent)
- *Dialogue* begins with PRO asserting an argument.
- Then PRO and OPP take turns in a *dispute*, where each player makes an argument that attacks his counterpart's last move
- Agent wins a dispute if his counterpart cannot counter attack
- But the counterpart may try a different line of attack, creating a new dispute
- This results in a *dispute tree* structure

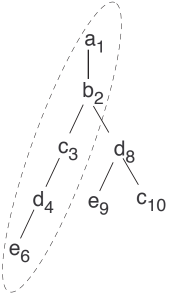
Example Dispute Tree



i)



ii)



iii)

i) an argumentation framework
ii) dispute tree induced in *a*
iii) dispute tree induced by *a* under protocol *G*; winning strategy circled

The Power of Abstract Argument Games

- We can say that a player A has a *winning strategy* for an argument *x* if, no matter what the other player does, player A wins.
- By adjusting the protocol, abstract argument games can correspond to different semantics:

  I.e. PRO wins if and only if the argument in question belongs to the corresponding extension.

- Adjustments include things like:
  - Preventing PRO or OPP from repeating their own moves or each others' moves
  - Preventing players from presenting arguments that attack arguments they themselves stated previously
  - etc.

The Power of Abstract Argument Games

- We can say that a player A has a *winning strategy* for an argument x if, no matter what the other player does, player A wins.
- By adjusting the protocol, abstract argument games can correspond to different semantics:

  I.e. PRO wins if and only if the argument in question belongs to the corresponding extension.

- Adjustments include things like:
  - Preventing PRO or OPP from repeating their own moves or each others' moves
  - Preventing players from presenting arguments that attack arguments they themselves stated previously
  - etc.

The Power of Abstract Argument Games

- We can say that a player A has a *winning strategy* for an argument x if, no matter what the other player does, player A wins.
- By adjusting the protocol, abstract argument games can correspond to different semantics:

  I.e. PRO wins if and only if the argument in question belongs to the corresponding extension.

- Adjustments include things like:
  - Preventing PRO or OPP from repeating their own moves or each others' moves
  - Preventing players from presenting arguments that attack arguments they themselves stated previously
  - etc.

Outline

What are Dialogue Systems?

- Due to philosopher Charles Hamblin
- Unlike abstract argument games, arguments have explicit internal structure
- Also typically more permissible than abstract argument games
- Used to study intricet dynamics of commitments

  Significant work by philosophers Douglas Walton and Eric Krabbe

- Can reveal fallacies in dialogue
- Uses elements of *speech act* theory (due to John Searle)

Elements of a Dialogue System

Due to Prakken

Speech acts with the possible responses (attack and surrender)

Acts	**Intuitive Meaning**	**Attacks**	**Surrenders**
claim φ	Assert φ is true	*why* φ	*concede* φ
φ *since* S	Support φ by argument S	*why* $\psi(\psi \in S)$ φ' *since* S' (defeats φ *since* S)	*concede* ψ $(\psi \in S)$ *concede* φ
why φ	Challenge φ	φ *since* S	*retract* φ
concede φ	Concede φ claimed by other		
retract φ	Take back own claim φ		

Dialogue System

- The protocol generates a tree structure
- Each utterance (i.e. move) is a node, and its possible responses are its children.
- Termination is defined in terms of the *dialogical status* of each move.
- A move is *in* if it is surrendered or if all its attacking replies are *out*.
- A move is *out* if it has a reply that is *in*.
- Whether the proponent or the opponent wins depends on whether the initial move is in or out.

Example

- Agent P has knowledge base $\{p, p \rightarrow_{r_1} q, p \rightarrow_{r_2} r, p \wedge s \rightarrow_{r_3} r_2 > r_4\}$
- Agent O has knowledge base $\{t, t \rightarrow_{r_4} \neg r\}$

This dialogue is consistent with the above protocol (target of each move indicated between square brackets):

$P_1[-]$:	*claim r*	$O_2[P_1]$:	*why r*
$P_3[O_2]$:	r *since* $q, q \Rightarrow r$	$O_4[P_3]$:	*why q*
$P_5[O_4]$:	q *since* $p, p \Rightarrow q$	$O_6[P_5]$:	*concede* $p \Rightarrow q$
		$O_7[P_5]$:	*why p*

Now, P can either:

- Retract his claim or premises of his argument, or
- give an argument in support of p.

And so on ...

Outline

Strategic Behaviour in Argumentation

- So far, we focused on protocols (e.g. dialogue systems)
- Protocols specify the set of possible moves agents can make
- Agents may have many choices about what to say at a given time
- These choices, the agent's *strategy*, significantly influence :
  1. the outcome of the dialogue

     (e.g. who wins)
  2. dialogue dynamics

     (e.g. whether it will terminate in a short number of moves).

Heuristic Argumentation Strategies

For example specify *attitudes* (due to Parsons et al, JLC):

- Assertion attitudes:
  - *confident* agent asserts any proposition for which he can construct an argument,
  - *careful* agent can do so only if he can construct such an argument and cannot construct a stronger counterargument
  - *thoughtful* agent can assert an a proposition only if he can construct an acceptable argument for the proposition.
- Evaluation attitudes:
  - *credulous* agent accepts a proposition if he can construct an argument for it
  - *cautious* agent does so only if he's also unable to construct a stronger counterargument
  - *skeptical* agent accepts an argument only if he can construct an acceptable argument for the proposition

Heuristic Argumentation Strategies

For example specify *attitudes* (due to Parsons et al, JLC):

- Assertion attitudes:
  - *confident* agent asserts any proposition for which he can construct an argument,
  - *careful* agent can do so only if he can construct such an argument and cannot construct a stronger counterargument
  - *thoughtful* agent can assert an a proposition only if he can construct an acceptable argument for the proposition.
- Evaluation attitudes:
  - *credulous* agent accepts a proposition if he can construct an argument for it
  - *cautious* agent does so only if he's also unable to construct a stronger counterargument
  - *skeptical* agent accepts an argument only if he can construct an acceptable argument for the proposition

Heuristic Strategies vs. Game Theory

- Heuristics only address a sub-set of the possible strategies.
- On the other hand, game theory can achieve two goals:
    1. undertake precise analysis of interaction in particular strategic settings, with a view to predicting the outcome
    2. design rules of the game in such a way that self-interested agents behave in some desirable way (e.g. tell the truth); this is called *mechanism design*
- What game theory can do with argumentation:
    1. An agent may use game theory to analyse a given argumentative situation in order to choose the best strategy.
    2. We may use mechanism design to design the rules (e.g. argumentation protocol) in such a way as to promote good argumentative behaviour.

Heuristic Strategies vs. Game Theory

- Heuristics only address a sub-set of the possible strategies.
- On the other hand, game theory can achieve two goals:
  1. undertake precise analysis of interaction in particular strategic settings, with a view to predicting the outcome
  2. design rules of the game in such a way that self-interested agents behave in some desirable way (e.g. tell the truth); this is called *mechanism design*
- What game theory can do with argumentation:
  1. An agent may use game theory to analyse a given argumentative situation in order to choose the best strategy.
  2. We may use mechanism design to design the rules (e.g. argumentation protocol) in such a way as to promote good argumentative behaviour.

Outline

Agents and their Types

- Self-interested agents I.
- $\theta_i \in \Theta_i$ denotes the *type* of agent i
- Agent's preferences over *outcomes* $o \in \mathcal{O}$
- Agent's preferences can be expressed by a utility function $u_i(o, \theta_i)$ which depends onoutcome agent's type
- Agent i prefers outcome o_1 to o_2 when $u_i(o_1, \theta_i) > u_i(o_2, \theta_i)$.

Strategies and Outcomes

- $s_i(\theta_i) \in \Sigma_i$ is agent *i*'s *strategy* (we ommit type θ_i if clear)
- $s = (s_1(\theta_1), \ldots, s_I(\theta_I))$ is a *strategy profile*
- Let $s_{-i}(\theta_{-i}) = (s_1(\theta_i), \ldots, s_{i-1}(\theta_{i-1}), s_{i+1}(\theta_{i+1}), \ldots, s_I(\theta_I))$

  So we can write: $s = (s_i, s_{-i})$

- $u_i((s_i, s_{-i}), \theta_i)$ is the utility of agent *i* with type θ_i when all agents play strategies specified by strategy profile $(s_i(\theta_i), s_{-i}(\theta_{-i}))$. Similarly, we also define:

$$\theta_{-i} = (\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_I)$$

Nash Equilibrium

A strategy profile $s^* = (s_1^*, \ldots, s_I^*)$ is a *Nash equilibrium* if no agent has incentive to change its strategy, given that no other agent changes.

Nash Equilibrium

$$\forall i, \forall s_i', u_i(s_i^*, s_{-i}^*, \theta_i) \geq u_i(s_i', s_{-i}^*, \theta_i)$$

Dominant Strategy Equilibrium

A strategy s_i is said to be *dominant* if by playing it, the utility of agent i is maximized no matter what strategies the other agents play.

Dominant Strategy

A strategy s_i^* is *dominant* if

$$\forall s_{-i},\ \forall s_i',\ u_i(s_i^*, s_{-i}, \theta_i) \geq u_i(s_i', s_{-i}, \theta_i).$$

A *dominant-strategy equilibrium* is a strategy profile where each agent is playing a dominant strategy.

The Objective

Define the desirable outcome:

Social Choice Function

A *social choice function* is a rule $f : \Theta_1 \times \ldots \times \Theta_I \rightarrow \mathcal{O}$, that selects some outcome $f(\theta) \in \mathcal{O}$, given agent types $\theta = (\theta_1, \ldots, \theta_I)$.

Challenges:

- Agent types θ_i are privately known
- Agents may mis-report them!

Mechanism

A mechanism is just a game we design!

Mechanism

A *mechanism* $\mathcal{M} = (\Sigma, g(\cdot))$ defines the set of allowable strategies that agents can chose, with $\Sigma = \Sigma_1 \times \cdots \times \Sigma_I$ where Σ_i is the strategy set for agent i, and an outcome function $g(s)$ which specifies an outcome o for each possible strategy profile $s = (s_1, \ldots, s_I) \in \Sigma$.

Possible strategies may be to simply reveal types directly:

Direct-Revelation Mechanism

A *direct-revelation mechanism* is a mechanism in which $\Sigma_i = \Theta_i$ for all i, and $g(\theta) = f(\theta)$ for all $\theta \in \Theta$.

Equaly powerful as any mechanism (see *Revelation Principle*)

Mechanism

A mechanism is just a game we design!

Mechanism

A *mechanism* $\mathcal{M} = (\Sigma, g(\cdot))$ defines the set of allowable strategies that agents can chose, with $\Sigma = \Sigma_1 \times \cdots \times \Sigma_I$ where Σ_i is the strategy set for agent i, and an outcome function $g(s)$ which specifies an outcome o for each possible strategy profile $s = (s_1, \ldots, s_I) \in \Sigma$.

Possible strategies may be to simply reveal types directly:

Direct-Revelation Mechanism

A *direct-revelation mechanism* is a mechanism in which $\Sigma_i = \Theta_i$ for all i, and $g(\theta) = f(\theta)$ for all $\theta \in \Theta$.

Equaly powerful as any mechanism (see *Revelation Principle*)

Implementation

A mechanism *implements* social choice function f if the outcome induced by the mechanism (in equilibrium) is the same outcome that the social choice function would have returned if the true types of the agents were known.

Implementation

Mechanism $\mathcal{M} = (\Sigma, g(\cdot))$ *implements* social choice function f if there exists an equilibrium s^* s.t. $\forall \theta \in \Theta,\ g(s^*(\theta)) = f(\theta)$

Mechanism Design Problem

Design the game such that we obtain the desirable *outcome* (defined by the *social choice function*) even when agents may not be truthful?

Incentive Compatibility

The social choice function $f(\cdot)$ is *incentive compatible* (or *truthfully implementable*) if the direct mechanism $\mathcal{M} = (\Theta, g(\cdot))$ has an equilibrium s^* such that $s_i^*(\theta_i) = \theta_i$.

If the equilibrium is the dominant-strategy equilibrium, then the social choice function is *strategy-proof*

Outline

General Idea

- Agent *types* reflect:
  - The arguments they know
  - Their preferences over which arguments they'd like accepted
- *Strategies* define which arguments to assert
  - Agents may mis-report the arguments (e.g. hide evidence)
- Mechanism designer's *desirable* outcome is to make an informed decision:
  - Decide on which arguments are acceptable using some criterion (e.g. grounded extension)
  - Use all available arguments
- *Mechanism (game)* is the argumentation *protocol*
  - E.g. a dialogue system or abstract argumentation game
- *Direct mechanism* simply asks agents to reveal all their arguments in a single step

General Idea

- Agent *types* reflect:
  - The arguments they know
  - Their preferences over which arguments they'd like accepted
- *Strategies* define which arguments to assert
  - Agents may mis-report the arguments (e.g. hide evidence)
- Mechanism designer's *desirable* outcome is to make an informed decision:
  - Decide on which arguments are acceptable using some criterion (e.g. grounded extension)
  - Use all available arguments
- *Mechanism (game)* is the argumentation *protocol*
  - E.g. a dialogue system or abstract argumentation game
- *Direct mechanism* simply asks agents to reveal all their arguments in a single step

General Idea

- Agent *types* reflect:
  - The arguments they know
  - Their preferences over which arguments they'd like accepted
- *Strategies* define which arguments to assert
  - Agents may mis-report the arguments (e.g. hide evidence)
- Mechanism designer's *desirable* outcome is to make an informed decision:
  - Decide on which arguments are acceptable using some criterion (e.g. grounded extension)
  - Use all available arguments
- *Mechanism (game)* is the argumentation *protocol*
  - E.g. a dialogue system or abstract argumentation game
- *Direct mechanism* simply asks agents to reveal all their arguments in a single step

General Idea

- Agent *types* reflect:
  - The arguments they know
  - Their preferences over which arguments they'd like accepted
- *Strategies* define which arguments to assert
  - Agents may mis-report the arguments (e.g. hide evidence)
- Mechanism designer's *desirable* outcome is to make an informed decision:
  - Decide on which arguments are acceptable using some criterion (e.g. grounded extension)
  - Use all available arguments
- *Mechanism (game)* is the argumentation *protocol*
  - E.g. a dialogue system or abstract argumentation game
- *Direct mechanism* simply asks agents to reveal all their arguments in a single step

General Idea

- Agent *types* reflect:
  - The arguments they know
  - Their preferences over which arguments they'd like accepted
- *Strategies* define which arguments to assert
  - Agents may mis-report the arguments (e.g. hide evidence)
- Mechanism designer's *desirable* outcome is to make an informed decision:
  - Decide on which arguments are acceptable using some criterion (e.g. grounded extension)
  - Use all available arguments
- *Mechanism (game)* is the argumentation *protocol*
  - E.g. a dialogue system or abstract argumentation game
- *Direct mechanism* simply asks agents to reveal all their arguments in a single step

Abstract Argumentation as a Mechanism

MD Concept	ArgMD Instantiation
Agent type $\theta_i \in \Theta_i$	Agent's arguments $\theta_i = \mathcal{A}_i \subseteq \mathcal{A}$
Outcome $o \in \mathcal{O}$	Accepted arguments $Acc(.) \subseteq \mathcal{A}$
Utility $u_i(o, \theta_i)$	Preferences over $2^{\mathcal{A}}$ (what arguments end up being accepted)
Social choice function $f : \Theta_1 \times \ldots \times \Theta_I \to \mathcal{O}$	$f(\mathcal{A}_1, \ldots, \mathcal{A}_I) = Acc(\langle A_1 \cup \ldots \cup \mathcal{A}_I, \mathcal{R} \rangle, \mathcal{S})$. by some argument acceptability criterion
Mechanism $\mathcal{M} = (\Sigma, g(\cdot))$ where $\Sigma = \Sigma_1 \times \cdots \times \Sigma_I$ and $g : \Sigma \to \mathcal{O}$	Σ_i is an argumentation strategy, $g : \Sigma \to 2^{\mathcal{A}}$
Direct mechanism: $\Sigma_i = \Theta_i$	$\Sigma_i = 2^{\mathcal{A}}$ (every agent reveals a set of arguments)
Truth revelation	Revealing $\mathcal{A}_i$

Outline

A Mechanism based on Grounded Semantics

Take all arguments revealed by the agents, and compute the grounded extension.

Grounded Direct Argumentation Mechanism

A *grounded direct argumentation mechanism* for argumentation framework $\langle \mathcal{A}, \mathcal{R} \rangle$ is $\mathcal{M}_{AF}^{grnd} = (\Sigma_1, \ldots, \Sigma_I, g(.))$ where:

- $\Sigma_i \in 2^{\mathcal{A}}$ is the set of strategies available to each agent;
- $g : \Sigma_1 \times \cdots \times \Sigma_I \to 2^{\mathcal{A}}$ is an outcome rule defined as: $g(\mathcal{A}_1^\circ, \ldots, \mathcal{A}_I^\circ) = Acc(\langle \mathcal{A}_1^\circ \cup \cdots \cup \mathcal{A}_I^\circ, \mathcal{R} \rangle, \mathcal{S}^{grnd})$ where $\mathcal{S}^{grnd}$ denotes sceptical grounded acceptability semantics.

Illustrative Example

- Assume agents can only lie by hiding arguments (not by making up arguments).

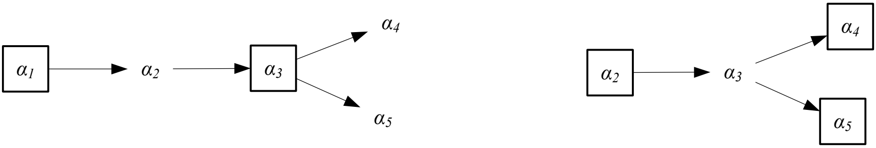
$$\forall i, \Sigma_i \in 2^{\mathcal{A}_i}$$

- Suppose every agent attempts to maximise the number of arguments in $\mathcal{A}_i$ that end up being accepted. We call this preference criteria the *individual acceptability maximising preference*.

$$\forall o_1, o_2 \in \mathcal{O} \textit{ such that } |o_1 \cap \mathcal{A}_i| \geq |o_2 \cap \mathcal{A}_i|\textit{, we have } u_i(o_1, \mathcal{A}_i) \geq u_i(o_2, \mathcal{A}_i).$$

Hiding an Argument can be Beneficial

Consider agents with types $\mathcal{A}_x = \{\alpha_1, \alpha_4, \alpha_5\}$, $\mathcal{A}_y = \{\alpha_2\}$ and $\mathcal{A}_z = \{\alpha_3\}$. Suppose defeat is as in figure (a). If each agent reveals everything, we accept the arguments shown in boxes (i.e. outcome is $o = \{\alpha_1, \alpha_3\}$). If agents have individual acceptability maximising preferences, with utilities equal to the number of arguments accepted, then: $u_x(o, \{\alpha_1, \alpha_4, \alpha_5\}) = 1$; $u_y(o, \{\alpha_3\}) = 1$; and $u_z(o, \{\alpha_2\}) = 0$.



(a) Argument graph in case of full revelation

(b) Argument graph with α_1 withheld

What hapens if agent x hides α_1? We get figure (b). Agent x benefits.

Condition in which Hiding is not Beneficial

Indirect Defeat

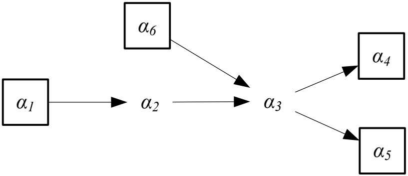
Let $\alpha, \beta \in \mathcal{A}$. We say that α *indirectly defeats* β, written $\alpha \hookrightarrow \beta$, if and only if there is an odd-length path from α to β in the argument graph.

Theorem [Rahwan and Larson]

Suppose agents have individual acceptability maximising preferences. If each agent's type corresponds to a conflict-free set of arguments which does not include (in)direct defeats (formally $\forall i \nexists \alpha_1, \alpha_2 \in \mathcal{A}_i$ such that $\alpha_1 \hookrightarrow \alpha_2$), then $\mathcal{M}_{AF}^{grnd}$ is strategy-proof.

Necessary but not Sufficient

Let the agent types be $\mathcal{A}_x = \{\alpha_1, \alpha_4, \alpha_5, \alpha_6\}$, $\mathcal{A}_y = \{\alpha_2\}$ and $\mathcal{A}_z = \{\alpha_3\}$ respectively. The full argument graph is depicted above. Under full revelation, the mechanism outcome rule produces the outcome $o = \{\alpha_1, \alpha_4, \alpha_5, \alpha_6\}$.



Truth revelation is now a dominant strategy for x (since it gets all its arguments accepted) despite the fact that $\alpha_1 \hookrightarrow \alpha_4$ and $\alpha_1 \hookrightarrow \alpha_5$.

Outline

Why Argument Interchange Format?

- Common language for annotating argument structures
- Enable export / import between argumentation support tools
- Ideally:
    - Expressive enough (but not too much)
    - Extensible / customizable
    - Implementable with standard ontology languages (allows using common parsers, or even Web 3.0 reasoning engines)

Elements of the Argument Interchange Format

- Nodes (vertices) of two main types:
  - *information nodes* (or *I-nodes*) $\mathcal{N}_I \subset \mathcal{N}$:

    represent a claim, premise, data, etc.
  - *scheme nodes* (or *S-nodes*) $\mathcal{N}_S \subset \mathcal{N}$:

    capture applications of patterns of reasoning

Connecting Information Nodes via Scheme Nodes

- Cannot connect two I-Nodes directly
- Must go via an S-Node, which captures the relationship
- Scheme nodes themselves can be of three types:
  - *rule of inference application nodes* (or *RA-nodes*):

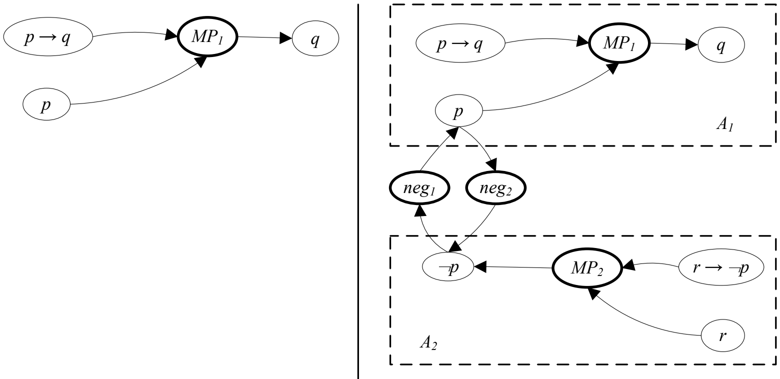
    e.g. an application of modus ponens
  - *preference application nodes* (or *PA-nodes*):

    e.g. an application of classical negation
  - *conflict application nodes* (or *CA-nodes*):

    e.g. statement of preference among two rules or two statements

AIF Example

S-Nodes denoted with a thicker border



(a) Simple argument

(b) Attack among two simple arguments

- MP_1 denotes an application of modus ponens
- neg_1 denotes an application of classical negation

AIF vs. Other Representations

- AIF is less abstract than Dung's abstract argument graphs
- But more abstract than the chained inference rules approach
- AIF was deliberately given only semi-formal semantics

  Can be adapted according to one's need, for example with a particular language for describing the internal contents of information nodes, or by committing to edges with specific formal semantics

- AIF is still a young effort in need of further refinement and proof-of-concept applications.

Some Implementations of AIF

- Prototype using the Resource Description Framework Schema (RDFS) [Zablith et al]
- Prototype using the Web Ontology Language (OWL) [Rahwan et al]

  Allows automated classification of schemes using Description Logic reasoning engines

- ArguBlogging [Snaith, Bex, Lawrence, Reed]

  Allows agreement and disagreement with any online content (news items, conversations etc.)

- Growing literature –see proceedings of the conference on Computational Models of Argument (COMMA)

For Further Reading I

- Douglas Walton
  Fundamentals of Critical Argumentation.
  Cambridge University Press, 2006.
- Iyad Rahwan and Guillermo R. Simari (Eds.)
  Argumentation in Artificial Intelligence.
  Springer, 2009.
- Philippe Besnard and Anthony Hunter
  Elements of Argumentation.
  MIT Press, 2008.
- Chris Reed and Tim J. Norman (Eds.)
  Argumentation Machines.
  Springer, 2004.