Knowledge Representation

Lecture 10: Discussion Games

*slides adapted from Martin Caminada at the Summer School for Argumentation in Cardiff

Atefeh Keshavarzi

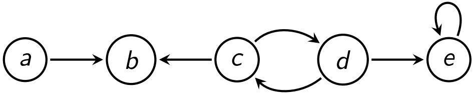
27, November 2023

Flashback: Dung's Abstract Argumentation Frameworks

Definition

An argumentation framework (AF) is a pair (A, R) where

- A is a set of arguments
- $R \subseteq A \times A$ is a relation representing the conflicts ("attacks")

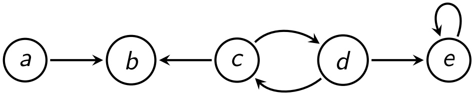


Flashback: Dung's Abstract Argumentation Frameworks

Definition

An argumentation framework (AF) is a pair (A, R) where

- A is a set of arguments
- $R \subseteq A \times A$ is a relation representing the conflicts ("attacks")

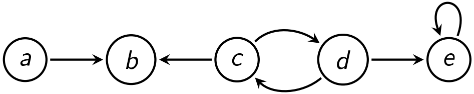


How can we assess the credibility of an argument in an AF?

An argument is believable if it can be argued successfully against the counterarguments.

- Semantics: Methods used to clarify the acceptance of arguments
  - Extension-based semantics
  - Labelling-based semantics

Flashback: Semantics of AFs



- $cf(F) = \{\{a, c\}, \{a, d\}, \{b, d\}, \{a\}, \{b\}, \{c\}, \{d\}, \{\}\}$
- $adm(F) = \{\{a, c\}, \{a, d\}, \{a\}, \{c\}, \{d\}, \{\}\}$
- $pref(F) = \{\{a, c\}, \{a, d\}\}$
- $stb(F) = \{\{a, d\}\}$
- $comp(F) = \{\{a, c\}, \{a, d\}, \{a\}\}$
- $grd(F) = \{\{a\}\}$

Labelling-based Semantics of AFs

Semantics of AFs

- Extension-based semantics
- Labelling-based semantics: The idea is to give each argument a label

Labelling-based Semantics of AFs

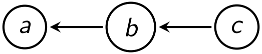
Semantics of AFs

- Extension-based semantics
- Labelling-based semantics: The idea is to give each argument a label

Definition

Given an AF $F = (A, R)$. A labelling is a function $\mathbb{L} : A \rightarrow \{\texttt{in}, \texttt{out}, \texttt{undec}\}$

- $\mathbb{L}(a) = \texttt{in}$, i.e., a is accepted;
- $\mathbb{L}(a) = \texttt{out}$, i.e., a is rejected;
- $\mathbb{L}(a) = \texttt{undec}$, i.e., a is undecided/unknown.



Labelling-based Semantics of AFs

Definition

Given an AF $F = (A, R)$. A labelling is a function $\mathbb{L} : A \to \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$

- $\mathbb{L}(a) = \mathtt{in}$, i.e., a is accepted;
- $\mathbb{L}(a) = \mathtt{out}$, i.e., a is rejected;
- $\mathbb{L}(a) = \mathtt{undec}$, i.e., a is undecided/unknown.

$$a \leftarrow b \leftarrow c$$

- $\mathbb{L}_1(A) = \{a \mapsto \mathtt{undec}, b \mapsto \mathtt{undec}, c \mapsto \mathtt{in}\}$

Labelling-based Semantics of AFs

Definition

Given an AF $F = (A, R)$. A labelling is a function $\mathbb{L} : A \to \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$

- $\mathbb{L}(a) = \mathtt{in}$, i.e., a is accepted;
- $\mathbb{L}(a) = \mathtt{out}$, i.e., a is rejected;
- $\mathbb{L}(a) = \mathtt{undec}$, i.e., a is undecided/unknown.

- $\mathbb{L}_2(A) = \{a \mapsto \mathtt{undec}, b \mapsto \mathtt{out}, c \mapsto \mathtt{in}\}$

Labelling-based Semantics of AFs

Definition

Given an AF $F = (A, R)$. A labelling is a function $\mathbb{L} : A \to \{\mathtt{in}, \mathtt{out}, \mathtt{undec}\}$

- $\mathbb{L}(a) = \mathtt{in}$, i.e., a is accepted;
- $\mathbb{L}(a) = \mathtt{out}$, i.e., a is rejected;
- $\mathbb{L}(a) = \mathtt{undec}$, i.e., a is undecided/unknown.

- $\mathbb{L}_3(A) = \{a \mapsto \mathtt{in}, b \mapsto \mathtt{out}, c \mapsto \mathtt{in}\}$

Labelling-based Semantics of AFs

Definition

Given an AF $F = (A, R)$. A labelling is a function $\mathbb{L} : A \rightarrow \{\texttt{in}, \texttt{out}, \texttt{undec}\}$

- $\mathbb{L}(a) = \texttt{in}$, i.e., a is accepted;
- $\mathbb{L}(a) = \texttt{out}$, i.e., a is rejected;
- $\mathbb{L}(a) = \texttt{undec}$, i.e., a is undecided/unknown.

- $\mathbb{L}_3(A) = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{in}\}$
- labelling-based argumentation semantics provides a way to select reasonable labellings among all the possible ones, according to some criterion.

Labelling-based Semantics of AFs (ctd.)

Each argument is labelled `in`, `out` or `undec`

an argument is `in` $\Leftrightarrow$
all its attackers are `out`

an argument is `out` $\Leftrightarrow$
it has an attacker that is `in`

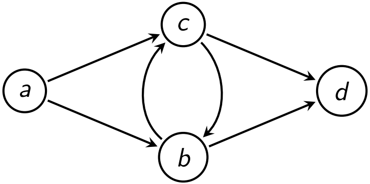
an argument is `undec` $\Leftrightarrow$
not all its attackers are `out` and it does not have an attacker that is `in`

Applying Argument Labellings

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

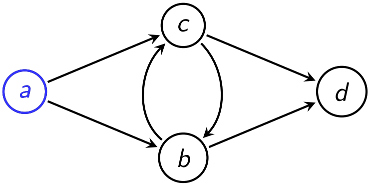


Applying Argument Labellings

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

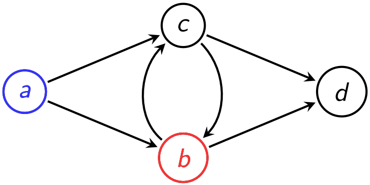


Applying Argument Labellings

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

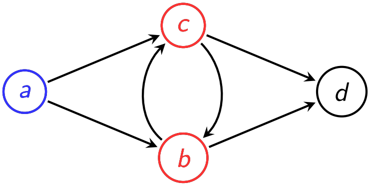


Applying Argument Labellings

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

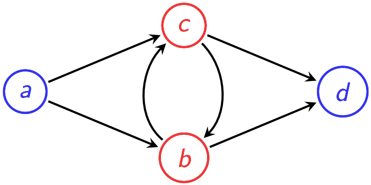


Applying Argument Labellings

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

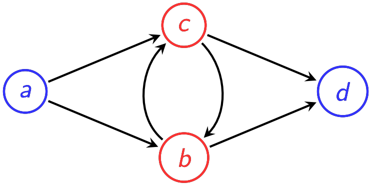


Applying Argument Labellings

$\texttt{in} \Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec} \Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$



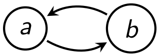
$$\mathbb{L}(A) = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{out}, d \mapsto \texttt{in}\}$$

Applying Argument Labellings (ctd.)

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

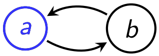


Applying Argument Labellings (ctd.)

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

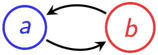


Applying Argument Labellings (ctd.)

$\texttt{in} \Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec} \Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$



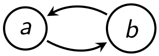
$\mathbb{L}_1(A) = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}\}$

Applying Argument Labellings (ctd.)

$\texttt{in}$ $\Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out}$ $\Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec}$ $\Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$

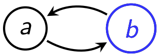


Applying Argument Labellings (ctd.)

in ⇔ all attackers are out

out ⇔ there is an attacker that is in

undec ⇔ not all attackers are out, and no attacker is in

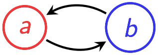


Applying Argument Labellings (ctd.)

$\texttt{in} \Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec} \Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$



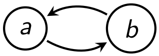
$$\mathbb{L}_2(A) = \{a \mapsto \texttt{out}, b \mapsto \texttt{in}\}$$

Applying Argument Labellings (ctd.)

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

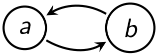


Applying Argument Labellings (ctd.)

$\mathtt{in} \Leftrightarrow$ all attackers are $\mathtt{out}$

$\mathtt{out} \Leftrightarrow$ there is an attacker that is $\mathtt{in}$

$\mathtt{undec} \Leftrightarrow$ not all attackers are $\mathtt{out}$, and no attacker is $\mathtt{in}$



$\mathbb{L}_1(A) = \{a \mapsto \mathtt{in}, b \mapsto \mathtt{out}\}$

$\mathbb{L}_2(A) = \{a \mapsto \mathtt{out}, b \mapsto \mathtt{in}\}$

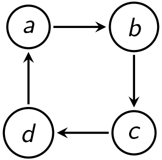
$\mathbb{L}_3(A) = \{a \mapsto \mathtt{undec}, b \mapsto \mathtt{undec}\}$

Applying Argument Labellings (ctd.)

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

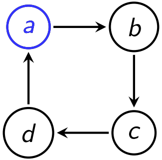


Applying Argument Labellings (ctd.)

in ⇔ all attackers are out

out ⇔ there is an attacker that is in

undec ⇔ not all attackers are out, and no attacker is in

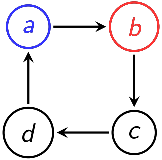


Applying Argument Labellings (ctd.)

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

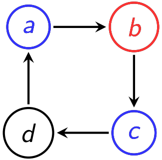


Applying Argument Labellings (ctd.)

in $\Leftrightarrow$ all attackers are out

out $\Leftrightarrow$ there is an attacker that is in

undec $\Leftrightarrow$ not all attackers are out, and no attacker is in

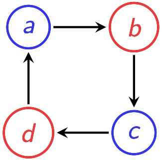


Applying Argument Labellings (ctd.)

$\texttt{in} \Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec} \Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$



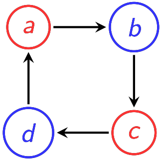
$$\mathbb{L}_1(A) = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{in}, d \mapsto \texttt{out}\}$$

Applying Argument Labellings (ctd.)

$\mathtt{in} \Leftrightarrow$ all attackers are $\mathtt{out}$

$\mathtt{out} \Leftrightarrow$ there is an attacker that is $\mathtt{in}$

$\mathtt{undec} \Leftrightarrow$ not all attackers are $\mathtt{out}$, and no attacker is $\mathtt{in}$



$$\mathbb{L}_2(A) = \{a \mapsto \mathtt{out}, b \mapsto \mathtt{in}, c \mapsto \mathtt{out}, d \mapsto \mathtt{in}\}$$

Applying Argument Labellings (ctd.)

$\texttt{in} \Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec} \Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$

$$\begin{array}{ccc} a & \rightarrow & b \\ \uparrow & & \downarrow \\ d & \leftarrow & c \end{array}$$

$$\mathbb{L}_3(A) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{undec}, d \mapsto \texttt{undec}\}$$

Applying Argument Labellings (ctd.)

$\texttt{in} \Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec} \Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$

$$\begin{array}{ccc} a & \rightarrow & b \\ \uparrow & & \downarrow \\ d & \leftarrow & c \end{array}$$

$\mathbb{L}_1(A) = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{in}, d \mapsto \texttt{out}\}$

$\mathbb{L}_2(A) = \{a \mapsto \texttt{out}, b \mapsto \texttt{in}, c \mapsto \texttt{out}, d \mapsto \texttt{in}\}$

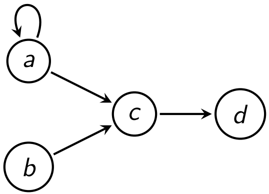
$\mathbb{L}_3(A) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{undec}, d \mapsto \texttt{undec}\}$

Applying Argument Labellings (ctd.)

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

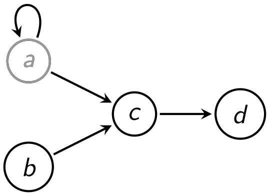


Applying Argument Labellings (ctd.)

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

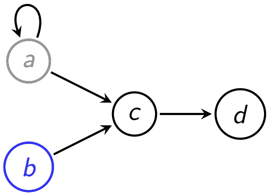


Applying Argument Labellings (ctd.)

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

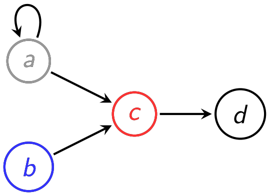


Applying Argument Labellings (ctd.)

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

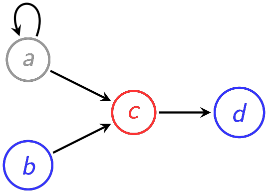


Applying Argument Labellings (ctd.)

$\texttt{in} \Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec} \Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$



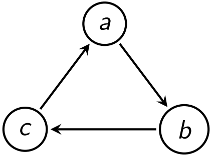
$$\mathbb{L}(A) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{in}, c \mapsto \texttt{out}, d \mapsto \texttt{in}\}$$

Applying Argument Labellings (ctd.)

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

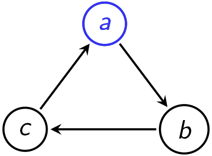


Applying Argument Labellings (ctd.)

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

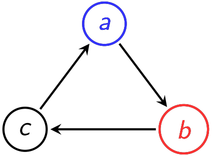


Applying Argument Labellings (ctd.)

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

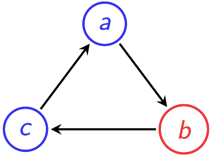


Applying Argument Labellings (ctd.)

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`

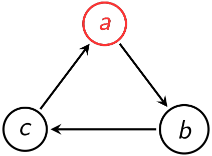


Applying Argument Labellings (ctd.)

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

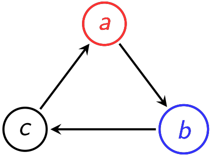


Applying Argument Labellings (ctd.)

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

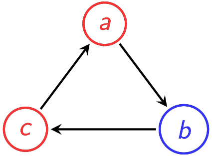


Applying Argument Labellings (ctd.)

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

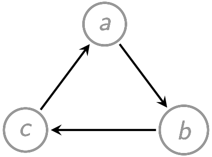


Applying Argument Labellings (ctd.)

in $\Leftrightarrow$ all attackers are out

out $\Leftrightarrow$ there is an attacker that is in

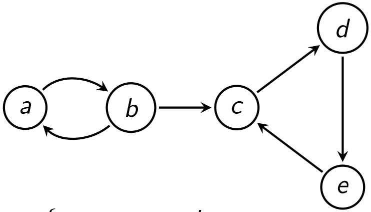
undec $\Leftrightarrow$ not all attackers are out, and no attacker is in



$\mathbb{L}(A) = \{a \mapsto \mathrm{undec}, b \mapsto \mathrm{undec}, c \mapsto \mathrm{undec}\}$

Applying Argument Labellings (ctd.)

Give the three labellings of this argumentation framework.



in $\Leftrightarrow$ all attackers are out

out $\Leftrightarrow$ there is an attacker that is in

undec $\Leftrightarrow$ not all attackers are out, and no attacker is in

1. $\mathbb{L}_1 = \{a \mapsto \texttt{undec}, b \mapsto \texttt{in}, c \mapsto \texttt{out}, d \mapsto \texttt{undec}, e \mapsto \texttt{out}\}$
2. $\mathbb{L}_2 = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{undec}, d \mapsto \texttt{undec}, e \mapsto \texttt{undec}\}$
3. $\mathbb{L}_3 = \{a \mapsto \texttt{out}, b \mapsto \texttt{in}, c \mapsto \texttt{undec}, d \mapsto \texttt{undec}, e \mapsto \texttt{undec}\}$
4. $\mathbb{L}_4 = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{undec}, d \mapsto \texttt{undec}, e \mapsto \texttt{undec}\}$
5. $\mathbb{L}_5 = \{a \mapsto \texttt{out}, b \mapsto \texttt{in}, c \mapsto \texttt{out}, d \mapsto \texttt{in}, e \mapsto \texttt{out}\}$

menti
.com
1684
4333

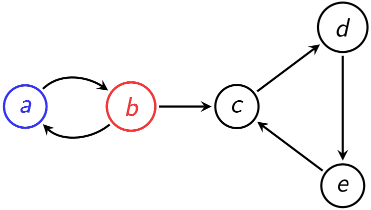
Applying Argument Labellings (ctd.)

Give the three labellings of this argumentation framework.

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\Leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`



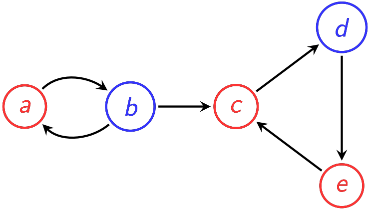
Applying Argument Labellings (ctd.)

Give the three labellings of this argumentation framework.

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`



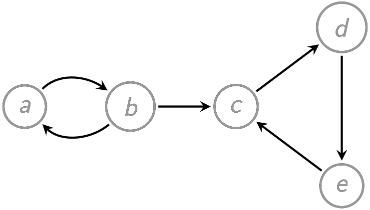
Applying Argument Labellings (ctd.)

Give the three labellings of this argumentation framework.

in $\Leftrightarrow$ all attackers are out

out $\Leftrightarrow$ there is an attacker that is in

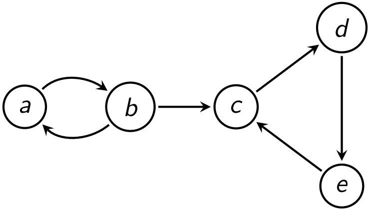
undec $\Leftrightarrow$ not all attackers are out, and no attacker is in



Maximisation/Minimisation

maximal: there is no other that has the same plus something

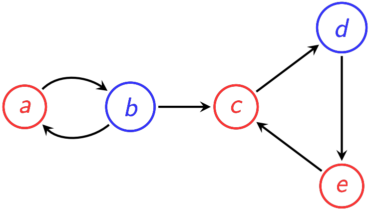
minimal: there is no other that has the same minus something



Maximisation/Minimisation

maximal: there is no other that has the same plus something

minimal: there is no other that has the same minus something

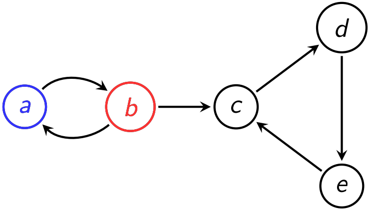


a, b, c, d, e max `in`, max `out`, min `undec`

Maximisation/Minimisation

maximal: there is no other that has the same plus something

minimal: there is no other that has the same minus something

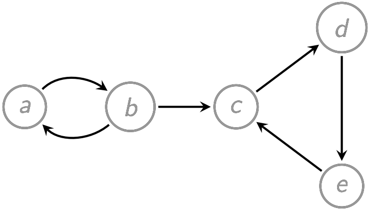


a, b, c, d, e max `in`, max `out`

Maximisation/Minimisation

maximal: there is no other that has the same plus something

minimal: there is no other that has the same minus something



a, b, c, d, e min `in`, min `out`, max `undec`

Extension-based Semantics vs. Labelling-based Semantics

`in` $\Leftrightarrow$ all attackers are `out`

`out` $\leftrightarrow$ there is an attacker that is `in`

`undec` $\Leftrightarrow$ not all attackers are `out`, and no attacker is `in`

restriction on labelling

maximal `in`

maximal `out`

maximal `undec`

minimal `in`

minimal `out`

empty `undec`

Extension-based Semantics vs. Labelling-based Semantics

in $\Leftrightarrow$ all attackers are out

out $\leftrightarrow$ there is an attacker that is in

undec $\Leftrightarrow$ not all attackers are out, and no attacker is in

restriction on labelling	Extension-based semantics
maximal in	preferred semantics
maximal out	preferred semantics
maximal undec	grounded semantics
minimal in	grounded semantics
minimal out	grounded semantics
empty undec	Stable semantics

An extension is the in-labelled part of a labelling

Admissible Labelling

Definition: Admissible labelling

Given an AF $F = (A, R)$. Let $\mathbb{L}$ be a labelling function on A. $\mathbb{L}$ is an admissible labelling iff for each argument $a \in A$ it holds that:

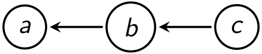
- if $\mathbb{L}(a) = \texttt{in}$ then for each b, such that $(b, a) \in R$ then $\mathbb{L}(b) = \texttt{out}$;
- if $\mathbb{L}(a) = \texttt{out}$ then there exists $b \in A$, such that $(b, a) \in A$ and $\mathbb{L}(b) = \texttt{in}$.

Admissible Labelling

Admissible labeling:

`in` $\Rightarrow$ all attackers are `out`

`out` $\Rightarrow$ there is an attacker that is `in`



Admissible Labelling

Admissible labeling:

in $\Rightarrow$ all attackers are out

out $\Rightarrow$ there is an attacker that is in

- $adm_{\mathbb{L}_1}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{undec}\}$

Admissible Labelling

Admissible labeling:

$\texttt{in} \Rightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Rightarrow$ there is an attacker that is $\texttt{in}$

- $adm_{\mathbb{L}_2}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{in}\}$

Admissible Labelling

Admissible labeling:

$\texttt{in} \Rightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Rightarrow$ there is an attacker that is $\texttt{in}$

- $adm_{\mathbb{L}_3}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{out}, c \mapsto \texttt{in}\}$

Admissible Labelling

Admissible labeling:

`in` $\Rightarrow$ all attackers are `out`

`out` $\Rightarrow$ there is an attacker that is `in`

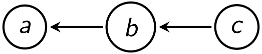
- $adm_{\mathbb{L}_4}(F) = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{in}\}$

Admissible Labelling

Admissible labeling:

$\texttt{in} \Rightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Rightarrow$ there is an attacker that is $\texttt{in}$



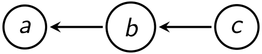
- $adm_{\mathbb{L}_1}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{undec}\}$
- $adm_{\mathbb{L}_2}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{in}\}$
- $adm_{\mathbb{L}_3}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{out}, c \mapsto \texttt{in}\}$
- $adm_{\mathbb{L}_4}(F) = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{in}\}$

Admissible Labelling

Admissible labeling:

$\texttt{in} \Rightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Rightarrow$ there is an attacker that is $\texttt{in}$



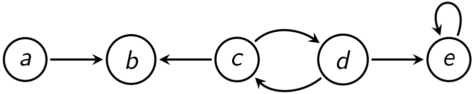
- $adm_{\mathbb{L}_1}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{undec}\}$ $adm_1(F) = \{\}$
- $adm_{\mathbb{L}_2}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{undec}, c \mapsto \texttt{in}\}$ $adm_2(F) = \{c\}$
- $adm_{\mathbb{L}_3}(F) = \{a \mapsto \texttt{undec}, b \mapsto \texttt{out}, c \mapsto \texttt{in}\}$ $adm_3(F) = \{c\}$
- $adm_{\mathbb{L}_4}(F) = \{a \mapsto \texttt{in}, b \mapsto \texttt{out}, c \mapsto \texttt{in}\}$ $adm_4(F) = \{a, c\}$

Complete Labelling

$\texttt{in} \Leftrightarrow$ all attackers are $\texttt{out}$

$\texttt{out} \Leftrightarrow$ there is an attacker that is $\texttt{in}$

$\texttt{undec} \Leftrightarrow$ not all attackers are $\texttt{out}$, and no attacker is $\texttt{in}$

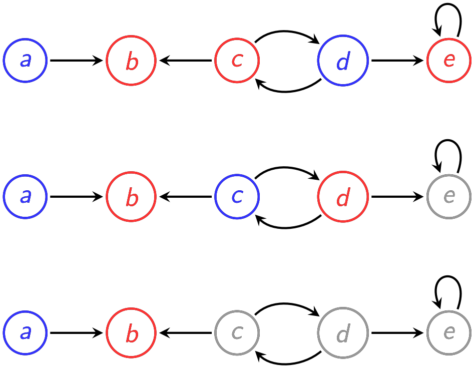


Complete Labelling

`in` ⇔ all attackers are `out`

`out` ⇔ there is an attacker that is `in`

`undec` ⇔ not all attackers are `out`, and no attacker is `in`



Labelling-based Semantics

Admissible Labelling

in $\Leftrightarrow$ all attackers are out

out $\Leftrightarrow$ there is an attacker that is in

Complete labelling

in $\Leftrightarrow$ all attackers are out

out $\Leftrightarrow$ there is an attacker that is in

undec $\leftrightarrow$ not all attackers are out, and no attacker is in

Grounded labelling: Complete with min in/ min out / max undec

Preferred labelling: Complete with max in/ max out

Stable labelling: complete with no undec

What is the relation between argumentation semantics and discussion?

- Preferred Discussion Game

Socratic Discussion

Answer me this. As soon as one man loves another, which of the two becomes the friend? the lover of the loved, or the loved of the lover? Or does it make no difference?

None in the world, that I can see

How? Are both friends, if only one loves?

I think so

Indeed! is it not possible for one who loves, not to be loved in return (. . .) ?

It is.

Nay, is it not possible for him even to be hated? (. . .) Don't you believe this to be true?

Quite true.

Well, in such a case as this, the one loves, the other is loved.

Just so.

Which of the two, then, is the friend of the other? The lover of the loved, whether or not he be loved in return, and even if he be hated, or the loved of the lover? or is neither the friend of he other, unless both love each other?

The latter certainly seems to be the case, Socrates.

If so, I continued, we think differently now from what we did before. (. . .)

Yes, I'm afraid we have contradicted ourselves.

Traditional Dialogue vs. Socratic Dialogue

- P: claim tr
  I think that there will be a tax relief.
- O: why tr
  Why do you think so?
- P: because pmp $\Rightarrow$ tr
  Because of the fact that the politicians made a promise.
- O: concede tr
  OK, you are right.

Traditional Dialogue vs. Socratic Dialogue

- P: claim tr
  I think that there will be a tax relief.
- O: but-then tr $\Rightarrow$ bd
  Then you implicitly also hold that budget deficit.
- P: concede bd
  Yes I do.
- O: but-then bd $\Rightarrow$ feu
  Then you implicitly also hold that fine from EU.
- P: concede feu
  Yes I do.
- O: but-then feu $\Rightarrow$ $\neg$tr
  Then you implicitly also hold that $\neg$tr.
- P: concede $\neg$tr *Oops, you're right; I caught myself in...*

because versus *but-then*

$$\top \ldots \underset{\substack{\longleftarrow \\ \text{because}}}{\Longrightarrow} \underset{\substack{\longleftarrow \\ \text{because}}}{\Longrightarrow} \underset{\substack{\longleftarrow \\ \text{because}}}{\Longrightarrow} \Phi$$

reasoning goes backward

proponent constructs path

originates from true

both parties become committed

$$\Phi \underset{\substack{\longrightarrow \\ \text{but-then}}}{\Longrightarrow} \underset{\substack{\longrightarrow \\ \text{but-then}}}{\Longrightarrow} \underset{\substack{\longrightarrow \\ \text{but-then}}}{\Longrightarrow} \ldots \bot$$

reasoning goes forward

opponent constructs path

leads to false

only proponent becomes committed

Preferred Semantics as Socratic Discussion

Recall: Definition

Admissible labeling:

- if argument is `in` then all attackers are `out`
- if argument is `out` then it has an attacker that is `in`

Recall: Proposition

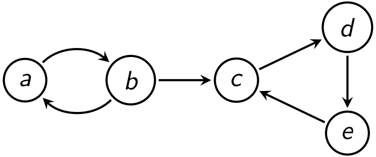
An argument is in a preferred extension
iff it is in a complete extension
iff it is in an admissible set
iff it is labelled `in` by an admissible labelling

Flashback: Decision Problems on AFs

Credulous Acceptance

Given an AF $F = (A, R)$, $a \in A$, and $\sigma \in \{adm, pref, stb, grd, comp, cf\}$.
$Cred_\sigma(a, F)$: is a contained in at least one σ-extension of F?

$$Cred_\sigma(a, F) = \begin{cases} \text{yes} & \text{if } \exists S \in \sigma\text{-extension } F \text{ s.t. } \; a \in S, \\ \text{no} & \text{otherwise} \end{cases}$$



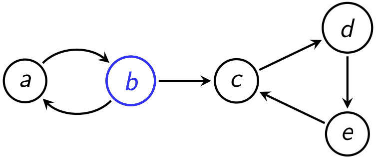
- $Cred_{adm}(b, F)$: is b contained in at least one adm-extension of F?

Flashback: Decision Problems on AFs

Credulous Acceptance

Given an AF $F = (A, R)$, $a \in A$, and $\sigma \in \{adm, pref, stb, grd, comp, cf\}$.
$Cred_\sigma(a, F)$: is a contained in at least one σ-extension of F?

$$Cred_\sigma(a, F) = \begin{cases} \text{yes} & \text{if } \exists S \in \sigma\text{-extension } F \text{ s.t. } \; a \in S, \\ \text{no} & \text{otherwise} \end{cases}$$



$adm(F) = \{\{b\}, \quad Cred_{adm}(b, F)$: Yes

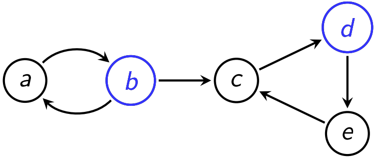
- $Cred_{adm}(b, F)$: is b contained in at least one adm-extension of F?

Flashback: Decision Problems on AFs

Credulous Acceptance

Given an AF $F = (A, R)$, $a \in A$, and $\sigma \in \{adm, pref, stb, grd, comp, cf\}$.
$Cred_\sigma(a, F)$: is a contained in at least one σ-extension of F?

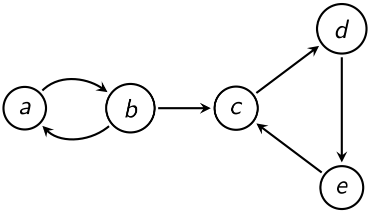
$$Cred_{cf}(a, F) = \begin{cases} \text{yes} & \text{if } (a, a) \notin R, \\ \text{no} & \text{otherwise} \end{cases}$$



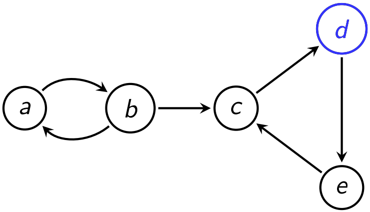
$pref(F) = \{\{b, d\},\quad Cred_{pref}(b, F)$: Yes

- $Cred_{pref}(b, F)$: is b contained in at least one *pref*-extension of F? Yes

Preferred Semantics as Socratic Discussion

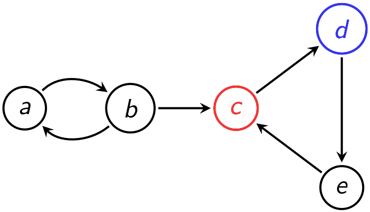


Preferred Semantics as Socratic Discussion



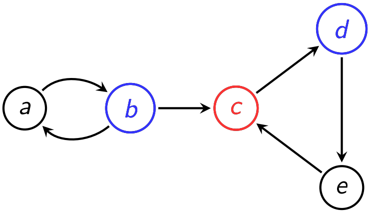
- P: `in`(d): I have an admissible labelling in which d is labelled `in`

Preferred Semantics as Socratic Discussion



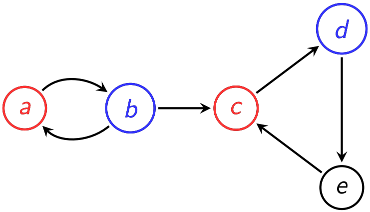
- P: `in`(d): I have an admissible labelling in which d is labelled `in`
- O: `out`(c): But then in your labelling it must also be the case that d's attacker c is labelled `out`. Based on which grounds?

Preferred Semantics as Socratic Discussion



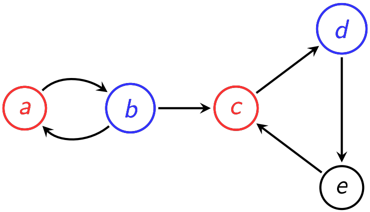
- P: in(d): I have an admissible labelling in which d is labelled in
- O: out(c): But then in your labelling it must also be the case that d's attacker c is labelled out. Based on which grounds?
- P: in(b): c is labelled out because b is labelled in.

Preferred Semantics as Socratic Discussion



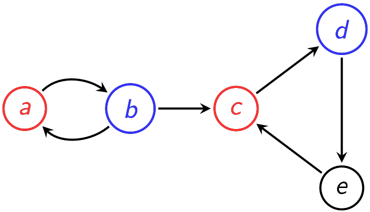
- P: `in`(d): I have an admissible labelling in which d is labelled `in`
- O: `out`(c): But then in your labelling it must also be the case that d's attacker c is labelled `out`. Based on which grounds?
- P: `in`(b): c is labelled `out` because b is labelled `in`.
- O: `out`(a): But then in your labelling it must also be the case that b's attacker a is labelled `out`. Based on which grounds?

Preferred Semantics as Socratic Discussion



- P: in(d): I have an admissible labelling in which d is labelled in
- O: out(c): But then in your labelling it must also be the case that d's attacker c is labelled out. Based on which grounds?
- P: in(b): c is labelled out because b is labelled in.
- O: out(a): But then in your labelling it must also be the case that b's attacker a is labelled out. Based on which grounds?
- P: in(b): a is labelled out because b is labelled in.

Preferred Semantics as Socratic Discussion



- P: in(d): I have an admissible labelling in which d is labelled in
- O: out(c): But then in your labelling it must also be the case that d's attacker c is labelled out. Based on which grounds?
- P: in(b): c is labelled out because b is labelled in.
- O: out(a): But then in your labelling it must also be the case that b's attacker a is labelled out. Based on which grounds?
- P: in(b): a is labelled out because b is labelled in.
  If O cannot make a move any more, P wins the discussion.

Preferred Semantics as Socratic Discussion

- P: `in`(d): I have an admissible labelling in which d is labelled `in`
- O: `out`(c): But then in your labelling it must also be the case that d's attacker c is labelled `out`. Based on which grounds?
- P: `in`(b): c is labelled `out` because b is labelled `in`.
- O: `out`(a): But then in your labelling it must also be the case that b's attacker a is labelled `out`. Based on which grounds?
- P: `in`(b): a is labelled `out` because b is labelled `in`.

1. Each move of P (except the first) contains an attacker of the directly preceding move of O.

Preferred Semantics as Socratic Discussion

- P: `in`(d): I have an admissible labelling in which d is labelled `in`
- O: `out`(c): But then in your labelling it must also be the case that d's attacker c is labelled `out`. Based on which grounds?
- P: `in`(b): c is labelled `out` because b is labelled `in`.
- O: `out`(a): But then in your labelling it must also be the case that b's attacker a is labelled `out`. Based on which grounds?
- P: `in`(b): a is labelled `out` because b is labelled `in`.

2. Each move of O contains an attacker of <u>some</u> previous move of P.

Preferred Semantics as Socratic Discussion

- P: in(d): I have an admissible labelling in which d is labelled in
- O: out(c): But then in your labelling it must also be the case that d's attacker c is labelled out. Based on which grounds?
- P: in(b): c is labelled out because b is labelled in.
- O: out(a): But then in your labelling it must also be the case that b's attacker a is labelled out. Based on which grounds?
- P: in(b): a is labelled out because b is labelled in.

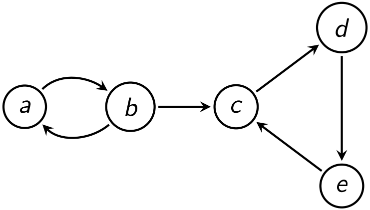
3. O is not allowed to repeat his moves.

Preferred Semantics as Socratic Discussion

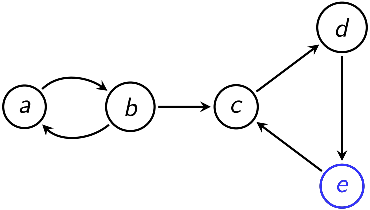
- P: in(d): I have an admissible labelling in which d is labelled in
- O: out(c): But then in your labelling it must also be the case that d's attacker c is labelled out. Based on which grounds?
- P: in(b): c is labelled out because b is labelled in.
- O: out(a): But then in your labelling it must also be the case that b's attacker a is labelled out. Based on which grounds?
- P: in(b): a is labelled out because b is labelled in.

1. Each move of P (except the first) contains an attacker of the directly preceding move of O.
2. Each move of O contains an attacker of some previous move of P.
3. O is not allowed to repeat his moves.
4. P is allowed to repeat his moves.

Preferred Semantics as Socratic Discussion

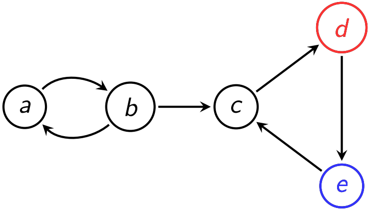


Preferred Semantics as Socratic Discussion



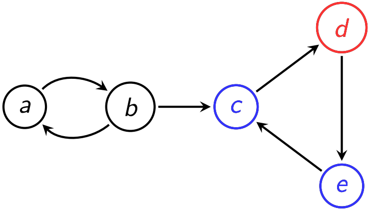
- P: $\texttt{in}(e)$: I have an admissible labelling in which e is labelled $\texttt{in}$

Preferred Semantics as Socratic Discussion



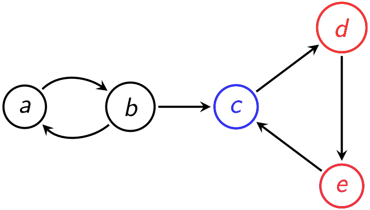
- P: in(e): I have an admissible labelling in which e is labelled in
- O: out(d): But then in your labelling it must also be the case that e's attacker d is labelled out. Based on which grounds?

Preferred Semantics as Socratic Discussion



- P: $\mathtt{in}(e)$: I have an admissible labelling in which e is labelled $\mathtt{in}$
- O: $\mathtt{out}(d)$: But then in your labelling it must also be the case that e's attacker d is labelled out. Based on which grounds?
- P: $\mathtt{in}(c)$: d is labelled $\mathtt{out}$ because c is labelled $\mathtt{in}$.

Preferred Semantics as Socratic Discussion



- P: in(e): I have an admissible labelling in which e is labelled in
- O: out(d): But then in your labelling it must also be the case that e's attacker d is labelled out. Based on which grounds?
- P: in(c): d is labelled out because c is labelled in.
- O: out(e): But then in your labelling it must also be the case that c's attacker e is labelled out.
  This **contradicts** with your earlier claim that e is labelled in

Preferred Semantics as Socratic Discussion

- P: in(e): I have an admissible labelling in which e is labelled in
- O: out(d): But then in your labelling it must also be the case that e's attacker d is labelled out. Based on which grounds?
- P: in(c): d is labelled out because c is labelled in.
- O: out(e): But then in your labelling it must also be the case that c's attacker e is labelled out.
  This **contradicts** with your earlier claim that e is labelled in
- **If O uses an argument previously used by P, then O wins the discussion.**

Preferred Semantics as Socratic Discussion

Winning rules

1. If O uses an argument previously used by P, then O **wins the discussion**.

Preferred Semantics as Socratic Discussion

Winning rules

2. If P uses an argument previously used by O, then O **wins the discussion**.

Preferred Semantics as Socratic Discussion

Winning rules

3. If P cannot make a move any more, O **wins the discussion**.

Preferred Semantics as Socratic Discussion

Winning rules

1. If O uses an argument previously used by P, then O **wins the discussion**.
2. If P uses an argument previously used by O, then O **wins the discussion**.
3. If P cannot make a move any more, O **wins the discussion**.
4. If O cannot make a move any more, P **wins the discussion**.

Preferred Semantics as Socratic Discussion

Theorem

Argument a is labelled `in` by at least one admissible labelling iff M can win the Socratic discussion game (for a).

Theorem

Argument a is labelled `in` by at least one preferred labelling iff M can win the Socratic discussion game (for a).

Why These Results Matter

- argumentation: based on notion of justification (entails what can be defended in rational discussion)
- discussions can be used by the system to explain its answer to the user
- different semantics express different types of rational discussion
- allows (in principle) for dynamic and user-based updating of the underlying knowledge base