An Introduction to Computational Argumentation
Semantics (5/5)
Topic: Structured Argumentation

Srdjan Vesic and Dragan Doder

ESSAI 2024

Structured Argumentation

- Abstract argumentation:
  - takes as given a set of arguments and attacks
  - provides semantics
- Structured argumentation:
  - provides a model of the structure and origin of arguments and attacks
  - allows to construct/derive arguments
    (e.g. from a knowledge base using rules of inference)

Structured Argumentation

- Abstract argumentation:
  - takes as given a set of arguments and attacks
  - provides semantics
- Structured argumentation:
  - provides a model of the structure and origin of arguments and attacks
  - allows to construct/derive arguments
    (e.g. from a knowledge base using rules of inference)
    - Arguments are reasons for conclusions
    - Infer conclusions from premises using *inference rules*

Structured Argumentation

- Abstract argumentation:
  - takes as given a set of arguments and attacks
  - provides semantics
- Structured argumentation:
  - provides a model of the structure and origin of arguments and attacks
  - allows to construct/derive arguments
    (e.g. from a knowledge base using rules of inference)
    - Arguments are reasons for conclusions
    - Infer conclusions from premises using *inference rules*
  - There are myriad systems out there, we will focus on a simple version of the popular ASPIC+ framework.

Syntax of arguments - ASPIC+

- Arguments are trees
  - Nodes are *formulas* of a logical language $\mathcal{L}$ (with negation $\neg$)

    (here we illustrate ASPIC+ using propositional language)
  - Links are applications of *inference rules*

    $\mathcal{R}_s$ – Strict rules ($\psi_1, \ldots, \psi_n \rightarrow \psi$)

    Strict rules cannot be challenged (if X is a penguin, then X is a bird)

    $\mathcal{R}_d$ – Defeasible rules ($\psi_1, \ldots, \psi_n \Rightarrow \psi$)

    Defeasible rules can be challenged (if X is a bird, it then it flies)

Syntax of arguments - ASPIC+

- Arguments are trees
  - Nodes are *formulas* of a logical language $\mathcal{L}$ (with negation $\neg$)

    (here we illustrate ASPIC+ using propositional language)
  - Links are applications of *inference rules*

    $\mathcal{R}_s$ – Strict rules $(\psi_1, \ldots, \psi_n \to \psi)$

    Strict rules cannot be challenged (if X is a penguin, then X is a bird)

    $\mathcal{R}_d$ – Defeasible rules $(\psi_1, \ldots, \psi_n \Rightarrow \psi)$

    Defeasible rules can be challenged (if X is a bird, it then it flies)
  - $n : \mathcal{R}_d \to \mathcal{L}$ is a naming convention for defeasible rules

Syntax of arguments - ASPIC+

- Arguments are trees
  - Nodes are *formulas* of a logical language $\mathcal{L}$ (with negation $\neg$)

    (here we illustrate ASPIC+ using propositional language)
  - Links are applications of *inference rules*

    $\mathcal{R}_s$ – Strict rules ($\psi_1, \ldots, \psi_n \rightarrow \psi$)

    Strict rules cannot be challenged (if X is a penguin, then X is a bird)

    $\mathcal{R}_d$ – Defeasible rules ($\psi_1, \ldots, \psi_n \Rightarrow \psi$)

    Defeasible rules can be challenged (if X is a bird, it then it flies)
  - $n : \mathcal{R}_d \rightarrow \mathcal{L}$ is a naming convention for defeasible rules
- *Reasoning* starts from a knowledge base $\mathcal{K} \subseteq \mathcal{L}$

  $\mathcal{K}_n$ – Necessary premises (*axioms*)

  $\mathcal{K}_p$ – Ordinary premises ("assumptions")

Arguments

Arguments are defined recursively

- ψ is an argument, if $\psi \in \mathcal{K}$;

Arguments

Arguments are defined recursively

- ψ is an argument, if $\psi \in \mathcal{K}$;
  - $Conc(A) = \psi$
  - $Prem(A) = \psi$
  - $Sub(A) = \{\psi\}$

Arguments

Arguments are defined recursively

- ψ is an argument, if $\psi \in \mathcal{K}$;
  - $Conc(A) = \psi$
  - $Prem(A) = \psi$
  - $Sub(A) = \{\psi\}$
- $A_1, \ldots, A_n \rightarrow / \Rightarrow \psi$ is an argument if
  - $A_1, \ldots, A_n$ are arguments
  - and there is some rule r $Conc(A_1), \ldots, Conc(A_n) \rightarrow / \Rightarrow \psi$

    If so
  - $Conc(A) = \psi$
  - $Prem(A) = Prem(A_1) \cup \ldots \cup Prem(A_n)$
  - $Sub(A) = Sub(A_1) \cup \ldots \cup Sub(A_n) \cup \{A\}$
  - We say that r is A's *top rule*

Exercise

Consider the knowledge base

$$\mathcal{K} = \{Bird, Pinguin\}$$

and the rule base

$$\mathcal{R}_d = \{r_1 : Bird \Rightarrow Flies, r_2 : Pinguin \Rightarrow \neg Flies, r_3 : Penguin \Rightarrow \neg n(r_1)\}$$

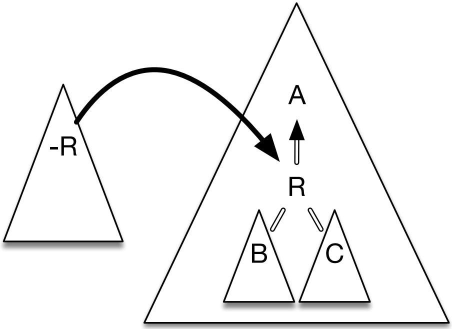
- Construct all the arguments that can be constructed using this knowledge base and rule base.

Types of attacks

- Undercutting: providing an exception to the rule
    - Attack the inference
- Undermining
    - Attack a premise (only an "assumption", not an axiom)
- Rebutting
    - Attack a conclusion (of a sub-argument with defeasible top rule)

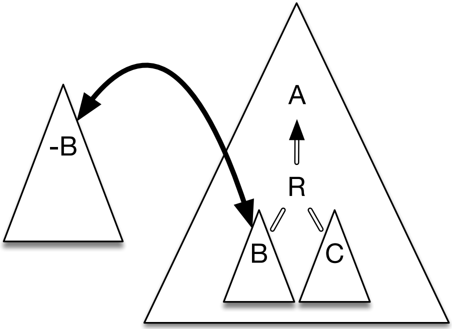
Undercuts

- Argument X undercuts an argument Y on Y' iff $Conc(X) = \neg n(r)$ for some $Y' \in Sub(Y)$ such that Y''s top rule r is defeasible.



Rebuts

- Argument X rebuts argument Y on Y' iff $Conc(X) = \neg Conc(Y')$ for some $Y' \in Sub(Y)$



Exercise

Consider the knowledge base

$$\mathcal{K} = \{Bird, Pinguin\}$$

and the rule base

$$\mathcal{R}_d = \{r_1 : Bird \Rightarrow Flies, r_2 : Pinguin \Rightarrow \neg Flies, r_3 : Penguin \Rightarrow \neg n(r_1)\}$$

- Construct all the arguments that can be constructed using this knowledge base and rule base.
- Indicate which of these arguments attack each other, and what the type of each attack is (rebut/undercut/undermine).

Exercise

Consider the knowledge base

$$\mathcal{K} = \{Bat, Baby\}$$

and the rule base

$$\mathcal{R} = \{r_1 : Bat \Rightarrow Flies, r_2 : Baby \Rightarrow \neg Flies, r_3 : Bat \rightarrow Mammal,$$
$$r_4 : Mammal \Rightarrow \neg Flies, r_5 : Baby \Rightarrow \neg n(r_1), r_6 : Bat \Rightarrow \neg n(r_4)\}$$

- Construct all the arguments that can be constructed using this knowledge base and rule base.
- Indicate which of these arguments attack each other, and what the type of each attack is (rebut/undercut/undermine).

Reasoning with structured argumentation

Process:

- From the knowledge base, generate arguments
- Identify attacks
- Evaluate using semantics
- Take the conclusions of the justified arguments

Reasoning with structured argumentation

Process:

- From the knowledge base, generate arguments
- Identify attacks
- Evaluate using semantics
- Take the conclusions of the justified arguments

- An argument *a* is sceptically justified if *a* belongs to all extensions (and there is at least one extension)
- An argument is credulously justified if it belongs to at least one extension, and it does not belong to all of them

Reasoning with structured argumentation

Process:

- From the knowledge base, generate arguments
- Identify attacks
- Evaluate using semantics
- Take the conclusions of the justified arguments

- An argument a is sceptically justified if a belongs to all extensions (and there is at least one extension)
- An argument is credulously justified if it belongs to at least one extension, and it does not belong to all of them

Exercise (cont.): What can we infer from the Penguin example?