Lecture 9: Structured Argumentation Frameworks

2-AIN-108 Computational Logic

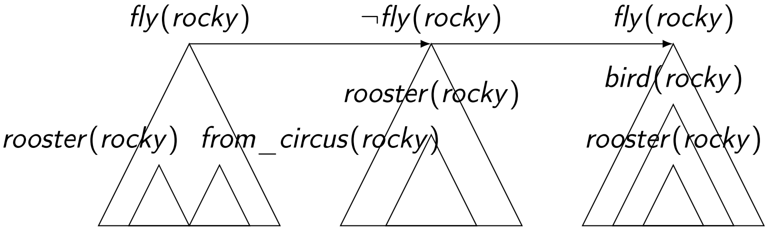
Martin Baláž, Martin Homola

Department of Applied Informatics
Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava

27 Nov 2012

Example

$$
\begin{array}{rcl}
bird(X) & \Rightarrow & fly(X) \\
rooster(X) & \rightarrow & bird(X) \\
rooster(X) & \Rightarrow & \neg fly(X) \\
rooster(X), from\_circus(X) & \Rightarrow & fly(X) \\
 & \rightarrow & rooster(rocky) \\
 & \rightarrow & from\_circus(rocky)
\end{array}
$$



Argumentation Process

1. Choosing an underlaying language
2. Constructing arguments
3. Identifying conflicts among arguments
4. Comparing arguments
5. Defining the status of arguments

Choosing an Underlaying Language

Definition (Strict and Defeasible Rule)

A strict rule is a formula of the form

$$L_1, \dots, L_n \rightarrow L_0$$

where $0 \leq n$ and each L_i, $0 \leq i \leq n$, is a classical literal.
A defeasible rule is a formula of the form

$$L_1, \dots, L_m, \sim L_{m+1}, \dots, \sim L_n \Rightarrow L_0$$

where $0 \leq m \leq n$ and each L_i, $0 \leq i \leq n$, is a classical literal.

Definition (Defeasible Logic Program)

A defeasible logic program is a finite set of strict or defeasible rules.

Constructing Arguments

Definition (Argument)

Let P be a defeasible logic program. An argument is

- a default argument $[L]$ where L is a default literal

$$\begin{aligned} Conc(A) &= L \\ SubArgs(A) &= \{A\} \end{aligned}$$

- a deductive argument $[A_1, \ldots, A_n \rightarrow / \Rightarrow L]$ if each A_i, $1 \leq i \leq n$, is an argument with $Conc(A_i) \models L_i$ and $L_1, \ldots, L_n \rightarrow / \Rightarrow L$ is a rule in P

$$\begin{aligned} Conc(A) &= L \\ SubArgs(A) &= SubArgs(A_1) \cup \cdots \cup SubArgs(A_n) \cup \{A\} \end{aligned}$$

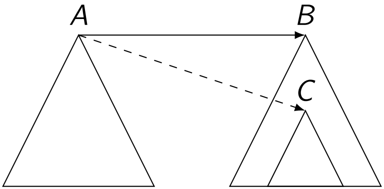
Identifying Conflicts among Arguments

Definition (Rebut)

An argument A rebuts an argument B (on its subargument C) iff C is a deductive subargument of B and $Conc(A) = \neg\, Conc(C)$.

Definition (Undercut)

An argument A undercuts an argument B (on its subargument C) iff C is a default subargument of B and $Conc(A) = \sim Conc(C)$.



Comparing Arguments

Preferences on rules

- Strict rules preferred over defeasible rules.
- Informations from more reliable source preferred over information from less reliable source.
- Newer information preferred over older information.
- ...

Preferences on arguments

- Arguments containing only strict rules are preferred over arguments containing a defeasible rule.
- Specific arguments preferred over general arguments.
- Arguments are compared with respect to the last defeasible rules.
- Arguments are compared with respect to all defeasible rules.
- ...

Defining the Status of Arguments

Definition (Argumentation Theory)

An argumentation theory for a defeasible logic program P is a pair $\mathcal{T} = (P, \prec)$ where $\prec \subseteq \mathcal{A} \times \mathcal{A}$ is a partial order on the set $\mathcal{A}$ of all arguments of P.

Definition (Argumentation Framework)

An argumentation framework for an argumentation theory $\mathcal{T} = (P, \prec)$ is a pair $\mathcal{F} = (\mathcal{A}, \mathcal{R})$ where $\mathcal{A}$ is the set of all arguments of P and $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ is an attack relation satisfying A attacks B iff A rebuts or undercuts B and $A \not\prec B$.

Now we can use any known semantics for abstract argumentation frameworks.

Example (Simplified)

$$
\begin{array}{ll}
A_1: & [\rightarrow \mathit{rooster}(\mathit{rocky})] \\
A_2: & [\rightarrow \mathit{from\_circus}(\mathit{rocky})] \\
A_3: & [A_1 \rightarrow \mathit{bird}(\mathit{rocky})] \\
A_4: & [A_3 \Rightarrow \mathit{fly}(\mathit{rocky})] \\
A_5: & [A_1 \Rightarrow \neg\, \mathit{fly}(\mathit{rocky})] \\
A_6: & [A_1, A_2 \Rightarrow \mathit{fly}(\mathit{rocky})]
\end{array}
$$

$$
\begin{array}{ccc}
A_4 & \prec & A_5 \\
A_5 & \prec & A_6
\end{array}
$$

Reasoning with Argumentation Frameworks

For an argument A of a finite argumentation framework $(\mathcal{A}, \mathcal{R})$:

- Is A contained in all extensions?
- Is A attacked by all extensions?
- Is A contained in an extension?
- Is A attacked by an extension?

Argumentation Game

Definition (Move)

A move is a pair $\mu = (P, A)$ where $P = pl(\mu) \in \{PRO, OPP\}$ and $A = arg(\mu)$ is an argument.

Definition (Legal Move Function)

A legal move function is a mapping $\phi\colon \mathcal{A}^+ \mapsto 2^{\mathcal{A}}$ where $\mathcal{A}^+$ is the set of all finite non-empty sequences of arguments.

Argumentation Game

Definition (Dialog)

Let ϕ be a legal move function. A ϕ-dialog is a non-empty sequence of moves $\mu_0, \mu_1, \ldots$ where

- $pl(\mu_0) = PRO$ and $pl(\mu_{i+1}) \neq pl(\mu_i)$
- $arg(\mu_{i+1}) \in \phi(arg(\mu_0), arg(\mu_1), \ldots, arg(\mu_i))$

Definition (Dialog Tree)

Let ϕ be a legal move function. A ϕ-dialog tree for an argument A is a minimal tree such that

- the root is $\mu_0 = (PRO, A)$
- if $\mu_0, \mu_1, \ldots, \mu_i$ is a path, $P \neq pl(\mu_i)$, and $A \in \phi(arg(\mu_0), arg(\mu_1), \ldots, arg(\mu_i))$, then $\mu_{i+1} = (P, A)$ is a child of μ_i.

Argumentation Game

Definition (Winning Dialog)

A ϕ-dialog d is won by PRO if $d = \mu_0, \mu_1, \ldots, \mu_i$ is finite, $\phi(arg(\mu_0), arg(\mu_1), \ldots, arg(\mu_i)) = \emptyset$, and $pl(\mu_i) = PRO$.

Definition (Proof)

A ϕ-proof for an argument A is a minimal finite subtree t' of ϕ-dialog tree t such that

- the root of t is the root of t'
- if μ is a node in t' and $pl(\mu) = PRO$ then all children of μ in t are also children of μ in t'
- if μ is a node in t' and $pl(\mu) = OPP$ then a child of μ in t which is also a child of μ in t'.
- all maximal branches of t' are won by PRO

Skeptical Complete (Grounded) Semantics

Definition (Legal Move Function Φ_G)

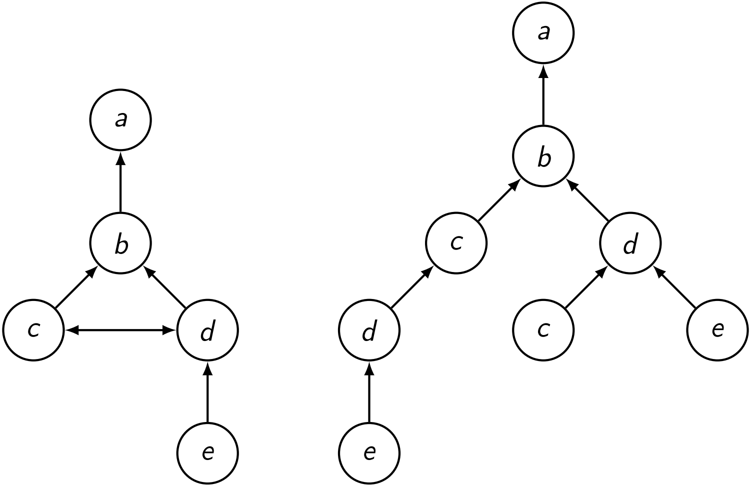
The legal move function ϕ_G is defined as follows:

- $A \in \phi_G(A_0, A_1, \ldots, A_{2i})$ iff
  - A attacks A_{2i}, i.e. $(A, A_{2i}) \in \mathcal{R}$
- $A \in \phi_G(A_0, A_1, \ldots, A_{2i+1})$ iff
  - A attacks A_{2i+1}, i.e. $(A, A_{2i+1}) \in \mathcal{R}$
  - *PRO* does not repeat arguments, i.e. $A_{2j} \neq A$ for all $0 \leq j \leq i$
  - *PRO* is conflict-free, i.e. $\{A_0, A_2, \ldots, A_{2i}, A\}$ is conflict-free

Proposition

1. *An argument A is in all complete extensions iff there exists ϕ_G-proof for A.*
2. *An argument A is in the grounded extension iff there exists ϕ_G-proof for A.*

Skeptical Complete (Grounded) Semantics



Credulous Complete (Preferred) Semantics

Definition (Legal Move Function Φ_P)

The legal move function ϕ_P is defined as follows:

- $A \in \phi_P(A_0, A_1, \ldots, A_{2i})$ iff
  - A attacks A_{2i}, i.e. $(A, A_{2i}) \in \mathcal{R}$
  - *OPP* does not repeat arguments, i.e. $A_{2j+1} \neq A$ for all $0 \leq j < i$
- $A \in \phi_P(A_0, A_1, \ldots, A_{2i+1})$ iff
  - A attacks A_{2i+1}, i.e. $(A, A_{2i+1}) \in \mathcal{R}$
  - *PRO* is conflict-free, i.e. $\{A_0, A_2, \ldots, A_{2i}, A\}$ is conflict-free

Proposition

1. *An argument A is in a complete extension iff there exists ϕ_P-proof for A.*
2. *An argument A is in a preferred extension iff there exists ϕ_P-proof for A.*

Credulous Complete (Preferred) Semantics