DECOMPOSING ABSTRACT DIALECTICAL FRAMEWORKS

Sarah Gaggl and Hannes Strass

Pitlochry, 12th September 2014
Motivation

- **Computational complexity** of semantics for ADFs is in general higher than for AFs [Strass and Wallner, 2014].
- Algorithms based on **SCC-recursive schema** for AF semantics show significant performance gain [Cerutti et.al. KR 2014].
- We propose a similar approach based on a recursive decomposition along SCCs.
- Allows to define \( cf_2 \) and \( stage_2 \) semantics for ADFs.
Motivation

- **Computational complexity** of semantics for ADFs is in general higher than for AFs [Strass and Wallner, 2014].
- Algorithms based on SCC-recursive schema for AF semantics show significant performance gain [Cerutti et.al. KR 2014].
- We propose a similar approach based on a recursive decomposition along SCCs.
- Allows to define $cf^2$ and $stage^2$ semantics for ADFs.

Main Difference to AFs

1. Acceptance conditions of statements in sub-frameworks may still depend on statements not contained in sub-framework.
2. Elimination of redundancies from links and acceptance formulas.
3. Propagation of truth values to subsequent SCCs.
Agenda

1. Introduction and Background
   - Abstract Dialectical Framework (ADFs)

2. Decomposing ADFs
   - Sub-Frameworks
   - Redundancies
   - Reduced Frameworks
   - Decomposition-based Semantics

3. Conclusion and Future Work
Like AFs, use graph to describe dependencies among nodes.
Unlike AFs, allow individual acceptance condition for each node.
Assigns \( t(\text{true}) \) or \( f(\text{false}) \) depending on status of parents.

**Definition**

An abstract dialectical framework (ADF) is a tuple \( D = (S, L, C) \) where

- \( S \) is a set of statements (positions, nodes),
- \( L \subseteq S \times S \) is a set of links,
- \( C = \{C_s\}_{s \in S} \) is a set of total functions \( C_s : 2^{\text{par}(s)} \rightarrow \{t, f\} \), one for each statement \( s \). \( C_s \) is called acceptance condition of \( s \).
Semantics

Definition

Let $\varphi$ be a propositional formula over vocabulary $S$ and for an $M \subseteq S$ let $v : M \to \{t, f, u\}$ be a three-valued interpretation. The partial valuation of $\varphi$ by $v$ is $\varphi^v = \varphi[p/t : v(p) = t][p/f : v(p) = f]$.

Definition

Let $D = (S, L, C)$ be an ADF. A three-valued interpretation $v$ is

- **conflict-free** iff for all $s \in S$ we have:
  - $v(s) = t$ implies that $\varphi^v_s$ is satisfiable,
  - $v(s) = f$ implies that $\varphi^v_s$ is unsatisfiable;
- **naive** iff it is $\leq_i$-maximal with respect to being conflict-free;

Where $\leq_i$ is a partial order over the truth values (resp. interpretations), i.e. $u <_i t$ and $u <_i f$.
Definition

Let $D = (S, L, C)$ be an ADF. The partial valuation of $\varphi$ by $v$ is $\varphi^v = \varphi[p/t : v(p) = t][p/f : v(p) = f]$.

A three-valued interpretation $v$ is

- **conflict-free** iff for all $s \in S$ we have:
  - $v(s) = t$ implies that $\varphi^v_s$ is satisfiable,
  - $v(s) = f$ implies that $\varphi^v_s$ is unsatisfiable;

- **naive** iff it is $\leq_i$-maximal with respect to being conflict-free;

Example

$v = \{a \mapsto f, b \mapsto u, c \mapsto t\}$ is conflict-free, as $\varphi^v_a = \neg t$ is unsatisfiable and $\varphi^v_c = \neg b$ is satisfiable.
Sub-Frameworks and Redundancies

- independent set $\text{ind}_D(\emptyset) = \{a, b, c\} = M_0$
- independent modulo $M_0$: $\text{ind}_D(M_0) = \{a, b, c, d, e, f\} = S$
- $M$ independent set: sub-framework $D|_M = (M, L \cap (M \times M), \{\varphi_s\}_{s \in M})$
Sub-Frameworks and Redundancies

- independent set $\text{ind}_D(\emptyset) = \{a, b, c\} = M_0$
- independent modulo $M_0$: $\text{ind}_D(M_0) = \{a, b, c, d, e, f\} = S$
- $M$ independent set: sub-framework $D|_M = (M, L \cap (M \times M), \{\varphi_s\}_{s \in M})$

- Redundancies can change dependencies between statements.
- If $(r, s)$ is redundant then $r$ has no influence on the truth value of $\varphi_s$ whatsoever.

Example

Consider $\varphi_s = a \lor (b \land c)$ and the interpretation $\nu = \{a \mapsto u, b \mapsto f, c \mapsto u\}$. $\varphi^\nu_s = a \lor (f \land c)$
Sub-Frameworks and Redundancies

- independent set $\text{ind}_D(\emptyset) = \{a, b, c\} = M_0$
- independent modulo $M_0$: $\text{ind}_D(M_0) = \{a, b, c, d, e, f\} = S$
- $M$ independent set: sub-framework $D|_M = (M, L \cap (M \times M), \{\varphi_s\}_{s \in M})$

- Redundancies can change dependencies between statements.
- If $(r, s)$ is redundant then $r$ has no influence on the truth value of $\varphi_s$ whatsoever.

Example

Consider $\varphi_s = a \lor (b \land c)$ and the interpretation $v = \{a \mapsto u, b \mapsto f, c \mapsto u\}$.
$\varphi^v_s = a \lor (f \land c) \equiv a$

$c$ has no influence
Reduced ADF

Given an ADF $D = (S, L, C)$, an independent set $M \subseteq S$ and an interpretation $v : M \rightarrow \{t, f, u\}$. The ADF $D$ reduced with $v$ on $M$ is obtained by:

- adapt the acceptance condition of statement $s$ to
  - $t$ (resp. $f$) if $v(s) = t$ (resp. $v(s) = f$)
  - $\neg s$ if $v(s) = u$
  - partial valuation $\varphi^v_s$ for remaining statements and if $r$ is redundant in $\varphi^v_s$, replace $r$ with $t$

- remove redundant links
- add links $\{(s, s) \mid v(s) = u\}$
Reduced ADF

Given an ADF $D = (S, L, C)$, an independent set $M \subseteq S$ and an interpretation $\nu : M \rightarrow \{t, f, u\}$. The ADF $D$ reduced with $\nu$ on $M$ is obtained by:

- adapt the acceptance condition of statement $s$ to
  - $t$ (resp. $f$) if $\nu(s) = t$ (resp. $\nu(s) = f$)
  - $\neg s$ if $\nu(s) = u$
  - partial valuation $\varphi^\nu_s$ for remaining statements and if $r$ is redundant in $\varphi^\nu_s$, replace $r$ with $t$

- remove redundant links
- add links $\{(s, s) \mid \nu(s) = u\}$

Procedure

For a semantics $\sigma$ and an ADF $D$, we obtain the $\sigma_2$ interpretations recursively by applying $\sigma_2(D) = \sigma_2(ind_D(\emptyset), D)$ by:

1. Start with all statements independent modulo $\emptyset$, i.e. $M_0 = ind_D(\emptyset)$
2. Compute all $\sigma$-interpretations of sub-framework $D|_{M_0}$
3. For each $\sigma$-interpretation $\nu$ of $D|_{M_0}$ compute the reduced ADF
4. Call Step 1 with reduced ADF and $M_1 = ind_D(M_0)$. 

Gaggl, Strass
Decomposing ADFs slide 12 of 27
Example

\[ \neg c \quad \neg a \quad \neg b \quad c \lor f \quad d \land f \quad e \]

\[ nai_2(D) = nai_2(\text{ind}_D(\emptyset), D) \]

\[ \text{ind}_D(\emptyset) = \{a, b, c\} = M_0 \]
$\neg c \quad \neg a \quad \neg b \quad c \lor f \quad d \land f \quad e$

$nai_2(D) = nai_2(ind_D(\emptyset), D)$

1. $ind_D(\emptyset) = \{a, b, c\} = M_0$
2. Then we obtain $nai(D|_{M_0}) = \{v_0, v_1, v_2\}$:

$v_0 = \{a \mapsto u, b \mapsto t, c \mapsto f\},$
$v_1 = \{a \mapsto f, b \mapsto u, c \mapsto t\},$
$v_2 = \{a \mapsto t, b \mapsto f, c \mapsto u\}.$
Example

\[ n_{ai2}(D) = n_{ai2}(ind_D(\emptyset), D) \]

1. \( ind_D(\emptyset) = \{a, b, c\} = M_0 \)
2. \( \nu_1 = \{a \mapsto f, b \mapsto u, c \mapsto t\} \)
Example

\[ nai_2(D) = nai_2(\text{ind}_D(\emptyset), D) \]

1. \[ \text{ind}_D(\emptyset) = \{a, b, c\} = M_0 \]
2. \[ \nu_1 = \{a \mapsto f, b \mapsto u, c \mapsto t\} \]
3. Reduced ADF with \( t \lor f \equiv t \), thus link \((f, d)\) is redundant
Example

\[ nai_2(D) = nai_2(ind_D(\emptyset), D) \]

1. \( ind_D(\emptyset) = \{a, b, c\} = M_0 \)
2. \( v_1 = \{a \mapsto f, b \mapsto u, c \mapsto t\} \)
3. Reduced ADF
Example

\[
\begin{align*}
\text{nai}_2(D) &= \text{nai}_2(\text{ind}_D(\emptyset), D) \\
\text{ind}_D(\emptyset) &= \{a, b, c\} = M_0 \\
v_1 &= \{a \mapsto f, b \mapsto u, c \mapsto t\} \\
\text{Reduced ADF} \\
\text{Call Step 1 with reduced ADF } D_1 \text{ and } M_1 = \text{ind}_{D_1}(M_0).
\end{align*}
\]
Example

\[ \text{nai}_2(M_1, D_1) \]

1. \[ M_1 = \text{ind}_{D_1}(M_0) = \{a, b, c, d\} \]
Example

\[ nai_2(M_1, D_1) \]

1. \( M_1 = \text{ind}_{D_1}(M_0) = \{a, b, c, d\} \)
2. \( nai(D_{|M_1}) = v_3 = v_1 \cup \{d \mapsto t\} \)
Example

\[ nai_2(M_1, D_1) \]

1. \( M_1 = \text{ind}_{D_1}(M_0) = \{a, b, c, d\} \)
2. \( nai(D|_{M_1}) = v_3 = v_1 \cup \{d \mapsto t\} \)
3. Reduced ADF
Example

\[ nai_2(M_1, D_1) \]

1. \[ M_1 = \text{ind}_{D_1}(M_0) = \{a, b, c, d\} \]
2. \[ nai(D|_{M_1}) = v_3 = v_1 \cup \{d \mapsto t\} \]
3. Reduced ADF
4. Call Step 1 with reduced ADF \( D_2 \) and \( M_2 = \text{ind}_{D_2}(M_1) \)
Example

\[ nai_2(M_2, D_2) \]

1. \[ M_2 = \text{ind}_{D_2}(M_1) = \{a, b, c, d, e, f\} = S \]
Example

\[ nai_2(M_2, D_2) \]

1. \[ M_2 = ind_{D_2}(M_1) = \{a, b, c, d, e, f\} = S \]
2. \[ nai(D_2) = \{v_4, v_6\}: \]

\[ v_4 = v_3 \cup \{e \mapsto \text{t}, f \mapsto \text{t}\}, \quad v_6 = v_3 \cup \{e \mapsto \text{f}, f \mapsto \text{f}\}. \]
### Main Theorem

<table>
<thead>
<tr>
<th>Theorem</th>
<th>Statement</th>
<th>Then</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Let $\sigma \in {\text{cfi, adm, pre, com, mod}}$.</td>
<td>$\sigma \leq \sigma_2$.</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>Let $\sigma \in {\text{nai, stg}}$.</td>
<td>$\sigma \nleq \sigma_2$.</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Let $\sigma \in {\text{cfi, nai, adm, pre, com, mod}}$.</td>
<td>$\sigma_2 \leq \sigma$.</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Let $\sigma \in {\text{stg}}$.</td>
<td>$\sigma_2 \nleq \sigma$.</td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

- We proposed a decomposition schema for semantics for ADFs.
- We introduced \( nai_2 \) and \( stg_2 \) for ADFs.
- Due to the relation of ADFs to logic programs we also get \( nai_2 \) and \( stg_2 \) semantics for LPs.
- Also in the paper: composing ADFs.

Future Work

- Analysis of the complexity of the approach.
- Implementation.
- Study splittings of ADFs and equivalences.
References

Leila Amgoud and Claudette Cayrol and Marie-Christine Lagasquie and Pierre Livet,
On Bipolarity in Argumentation Frameworks

AFRA: Argumentation Framework with Recursive Attacks.

Scc-recursiveness: A general schema for argumentation semantics.

G. Brewka and S. Woltran.
Abstract Dialectical Frameworks.

On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic
programming and n-person games.

S. Modgil.
Reasoning about Preferences in Argumentation Frameworks.

H. Strass and J. Wallner.
Analyzing the Computational Complexity of Abstract Dialectical Frameworks via Approximation
Fixpoint Theory.