Towards Preprocessing for Abstract Argumentation Frameworks

Thomas Linsbichler

Based on joint work with
Ringo Baumann, Wolfgang Dvořák and Stefan Woltran

Workshop on New Trends in Formal Argumentation
August 17th, 2017
Seminal Paper by Phan Minh Dung:

Example

\[ stb(F) = \{a, d, e\}, \]

Example

\[ stb(F) = \{\{a,d,e\},\{b,c,e\}\} \]
Seminal Paper by Phan Minh Dung:
On the acceptability of arguments and its fundamental role
in nonmonotonic reasoning, logic programming and n-person

Example

$$\text{stb}(F) = \{\{a, d, e\}, \{b, c, e\}\}$$
$$\text{pref}(F) = \{\{a, d, e\}\},$$
Seminal Paper by Phan Minh Dung:

Example

\[
\begin{align*}
stb(F) & = \{\{a, d, e\}, \{b, c, e\}\} \\
\text{pref}(F) & = \{\{a, d, e\}, \{b, c, e\}\}
\end{align*}
\]

Example

\[ stb(F) = \{\{a, d, e\}, \{b, c, e\}\} \]
\[ pref(F) = \{\{a, d, e\}, \{b, c, e\}, \{a, b\}\} \]
Prologue

Nonmon. Reasoning

Logic Programming

*n*-Person Games
Prologue

ABA

ASPIC

Nonmon. Reasoning

Logic Programming

$n$-Person Games

CoQuiAAS

ArgSemSAT

ASPARTIX

ConArg

Cegartix

Preprocessing

Thomas Linsbichler (TU Wien)  Preprocessing for Abstract Argumentation  August 17, 2017  3 / 31
Prologue

Preprocessing for Abstract Argumentation

Thomas Linsbichler (TU Wien)  Preprocessing for Abstract Argumentation  August 17, 2017  3 / 31
Prologue

Preprocessing for Abstract Argumentation

Thomas Linsbichler (TU Wien)  Preprocessing for Abstract Argumentation  August 17, 2017  3 / 31
Outline

- Quick Background on Argumentation Frameworks
- The Role of Preprocessing
- Theoretical Foundations
- Building a Preprocessing Machine
- Conclusions and Open Questions
Definition

An argumentation framework (AF) is a pair \((A, R)\) where \(A\) is a finite set of arguments and \(R \subseteq A \times A\) is the attack relation representing conflicts.

Semantics

For AF \(F = (A, R)\), \(E \in \sigma(F)\) iff . . .

- admissible: \(E\) is conflict-free and defends itself
- stable: \(E\) is conflict-free and has full range
- preferred: \(E\) is subset-maximal admissible
- complete: \(E\) is admissible and contains all defended arguments
- semi-stable: \(E\) is admissible with subset-maximal range
- stage: \(E\) is conflict-free with subset-maximal range
- grounded: \(E\) is subset-minimal complete set
- ideal: \(E\) is subset-maximal adm contained in each pref extension
<table>
<thead>
<tr>
<th>$\sigma$</th>
<th>$Cred_\sigma$</th>
<th>$Skept_\sigma$</th>
<th>$Ver_\sigma$</th>
<th>$NE_\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$cf$</td>
<td>in L</td>
<td>trivial</td>
<td>in L</td>
<td>in L</td>
</tr>
<tr>
<td>$grd$</td>
<td>P-c</td>
<td>P-c</td>
<td>P-c</td>
<td>in L</td>
</tr>
<tr>
<td>$stb$</td>
<td>NP-c</td>
<td>coNP-c</td>
<td>in L</td>
<td>NP-c</td>
</tr>
<tr>
<td>$adm$</td>
<td>NP-c</td>
<td>trivial</td>
<td>in L</td>
<td>NP-c</td>
</tr>
<tr>
<td>$comp$</td>
<td>NP-c</td>
<td>P-c</td>
<td>in L</td>
<td>NP-c</td>
</tr>
<tr>
<td>$ideal$</td>
<td>in $\Theta_2^P$</td>
<td>in $\Theta_2^P$</td>
<td>in $\Theta_2^P$</td>
<td>in $\Theta_2^P$</td>
</tr>
<tr>
<td>$pref$</td>
<td>NP-c</td>
<td>$\Pi_2^P$-c</td>
<td>coNP-c</td>
<td>NP-c</td>
</tr>
<tr>
<td>$sem$</td>
<td>$\Sigma_2^P$-c</td>
<td>$\Pi_2^P$-c</td>
<td>coNP-c</td>
<td>NP-c</td>
</tr>
<tr>
<td>$stage$</td>
<td>$\Sigma_2^P$-c</td>
<td>$\Pi_2^P$-c</td>
<td>coNP-c</td>
<td>in L</td>
</tr>
</tbody>
</table>
Background - ICCMA’17
(http://www.dbai.tuwien.ac.at/iccma17/)

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>D3</th>
<th>GO</th>
<th>PR</th>
<th>GT</th>
<th>DGT</th>
<th>GGT</th>
</tr>
</thead>
<tbody>
<tr>
<td>argmat_oLPB</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>argmat_dvisat</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>argmat-mpg</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>argmat-sat</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ArgSemSAT</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ArgTools</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ASPRMin</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Cogartix</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Chimaerarg</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CunArg</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CuQuiAA3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>EqArgSolver</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>gg-sts</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>goDIAMOND</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>heureka</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>pyglaf</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Preprocessing refers to a family of simplifications which are computationally easy to perform and are equivalence preserving

- SAT: tautology elimination, clause subsumption, ...

Proved very successful in SAT and QSAT solving

Example from the QBF world:
- Preprocessor Bloqger solved 471 of 1130 instances from QBFEVAL’16.
- DepQBF solves 556 instances without preprocessing, but 817 with preprocessing.

Preprocessing in the context of argumentation poses some additional challenges
**Preprocessing for Argumentation – Some Experiments**

Effect of (non equivalence-preserving) modifications with instances from the ICCMA’15 stable generator.

Thomas Linsbichler (TU Wien)  Preprocessing for Abstract Argumentation  August 17, 2017  9/31
Theoretical Foundations

In order to define possible preprocessing steps, we require
- a suitable notion of equivalence . . .
- which allows to verify which subparts of AFs can be simplified . . .
- under different semantics
Theoretical Foundations

In order to define possible preprocessing steps, we require

- a suitable notion of equivalence . . .
- which allows to verify which subparts of AFs can be simplified . . .
- under different semantics

More precisely, we want to find pairs \((F, F')\) such that replacing \(F\) by \(F'\) in any AF \(G\) does not change the extensions of \(G\)
Theoretical Foundations – Notions of Equivalence

Definition
Given a semantics \( \sigma \). Two AFs \( F \) and \( F' \) are (standard) equivalent w.r.t. \( \sigma \) (in symbols \( F \equiv \sigma F' \)) iff \( \sigma(F) = \sigma(F') \).

Definition
Given a semantics \( \sigma \). Two AFs \( F \) and \( F' \) are strongly equivalent w.r.t. \( \sigma \) (in symbols \( F \equiv^\sigma \sigma F' \)) iff \( F \cup H \equiv^\sigma F' \cup H \) holds for each AF \( H \).
**Example**

\[
\text{stb}(F) = \{\{a, d, e\}, \{b, c, e\}\} \quad \equiv \quad \text{stb}(F') = \{\{a, d, e\}, \{b, c, e\}\}
\]
Theoretical Foundations – Notions of Equivalence

Example

Follows from results in [Oikarinen & Woltran, 2011].
Theoretical Foundations – Notions of Equivalence

Example

\[ \text{pref}(F) = \{\{a, d, e\}, \{b, c, e\}, \{a, b\}\} \]

\[ \text{pref}(F') = \{\{a, d, e\}, \{b, c, e\}, \{a, b, e\}\} \]

\[ \neq \text{pref} \]
Theoretical Foundations – Notions of Equivalence

Example

\[ \text{stb}(F) = \{\{a, d, e\}, \{b, c, e\}\} \]

\[ \text{stb}(F') = \{\{a, d, e\}, \{b, c, e\}\} \]

\[ \text{stb}(F) \equiv_{\text{stb}} \text{stb}(F') \]
Theoretical Foundations – Notions of Equivalence

Example

\[ a \not\equiv_S b \]

\[ a \not\equiv_{stb} b \]

\[ e \]
Example

\[ stb(F \cup H) = \{\{a, d, e\}, \{b, c, e\}, \{a, b, e\}\} \]

but

\[ stb(F' \cup H) = \{\{a, d, e\}, \{b, c, e\}, \{a, b, e\}, \{a, d, f\}, \{b, c, f\}, \{a, b, f\}\} \]

\[ \neq_{stb} S \]
Observations:

- Standard equivalence is too weak for our purpose
- Strong equivalence is too restricted
  - For self-loop free AFs $F, F'$: $F \equiv_S F'$ iff $F = F'$!
Theoretical Foundations – Main Results

Observations:

- Standard equivalence is too weak for our purpose
- Strong equivalence is too restricted
  - For self-loop free AFs $F, F'$: $F \equiv_S F'$ iff $F = F'$!

We thus require a notion of equivalence which takes into account the neighborhood in an adequate way.

Definition

Given a semantics $\sigma$ and arguments $C \subseteq U$. Two AFs $F$ and $F'$ are $C$-relativized equivalent w.r.t. $\sigma$ (in symbols $F \equiv_C^{\sigma} F'$) iff $F \cup H \equiv^{\sigma} F' \cup H$ holds for each AF $H$ not containing arguments from $C$. 
Theoretical Foundations – Main Results

Observations:

- Standard equivalence is too weak for our purpose
- Strong equivalence is too restricted
  - For self-loop free AFs $F, F'$: $F \equiv_S F'$ iff $F = F'$!

We thus require a notion of equivalence which takes into account the neighborhood in an adequate way.

**Definition**

Given a semantics $\sigma$ and arguments $C \subseteq U$. Two AFs $F$ and $F'$ are $C$-relativized equivalent w.r.t. $\sigma$ (in symbols $F \equiv_C^{\sigma} F'$) iff $F \cup H \equiv^{\sigma} F' \cup H$ holds for each AF $H$ not containing arguments from $C$.

- for $C = \emptyset$, $C$-relativized equivalence coincides with strong equivalence
- for $C = U$, $C$-relativized equivalence is just standard equivalence
Definition

Given a semantics $\sigma$ and arguments $C \subseteq U$. Two AFs $F$ and $F'$ are $C$-relativized equivalent w.r.t. $\sigma$ (in symbols $F \equiv^C_\sigma F'$) iff $F \cup H \equiv^\sigma F' \cup H$ holds for each AF $H$ not containing arguments from $C$. 
Definition

Given a semantics $\sigma$ and arguments $C \subseteq U$. Two AFs $F$ and $F'$ are $C$-relativized equivalent w.r.t. $\sigma$ (in symbols $F \equiv^C_\sigma F'$) iff $F \cup H \equiv^\sigma C F' \cup H$ holds for each AF $H$ not containing arguments from $C$.

Example with $C = \{c, d, e, f\}$
Theoretical Foundations – Main Results

We first define a parameterized notion of the semantics.

**Definition**

Let $F = (A, R)$, $C \subseteq U$. The *C*-restricted stable extensions of $F$ are

$$stb_C(F) = \{ E \in cf(F) \mid A \cap C \subseteq E_F^\oplus \}$$
Theoretical Foundations – Main Results

We first define a parameterized notion of the semantics.

**Definition**

Let $F = (A, R)$, $C \subseteq U$. The $C$-restricted stable extensions of $F$ are

$$stb_C(F) = \{ E \in cf(F) \mid A \cap C \subseteq E_F^\oplus \}$$

**Example with $C = \{c, d, e, f\}$**

$$stb_C(F) = \{\{a, d, e\}, \{b, c, e\}, \{d, e\}, \{c, e\}\}$$

$$stb_C(F') = \{\{a, d, e\}, \{b, c, e\}, \{d, e\}, \{c, e\}\}$$
Theoretical Foundations – Main Results

We first define a parameterized notion of the semantics.

**Definition**

Let $F = (A, R)$, $C \subseteq U$. The $C$-restricted stable extensions of $F$ are

$$stb_C(F) = \{ E \in cf(F) \mid A \cap C \subseteq E^\oplus_F \}$$

**Example with $C = \{e, f\}$**

\[
\begin{align*}
stb_C(F) &= \{\{a, d, e\}, \{b, c, e\}, \\
&\quad \{d, e\}, \{c, e\}\} \\
stb_C(F') &= \{\{a, d, e\}, \{b, c, e\}, \\
&\quad \{d, e\}, \{c, e\}, \{a, e\}, \{b, e\}, \{e\}\}
\end{align*}
\]
For other semantics, such variants can be defined accordingly.

Definition

Let $F$ be an AF, $C \subseteq U$. We define

\[
\text{adm}_C(F) = \{ E \in \text{cf}(F) \mid E_F^- \cap C \subseteq E_F^+ \}
\]

\[
\text{pref}_C(F) = \{ E \in \text{adm}_C(F) \mid \text{for all } D \in \text{adm}_C(F) \text{ with } E \setminus C = D \setminus C, E_F^+ \setminus C \subseteq D_F^+ \setminus C, E_F^- \setminus E_F^+ \supseteq D_F^- \setminus D_F^+, E \cap C \not\subset D \cap C \}
\]
Theoretical Foundations – Main Results

For other semantics, such variants can be defined accordingly.

**Definition**

Let $F$ be an AF, $C \subseteq U$. We define

\[
adm_C(F) = \{ E \in cf(F) \mid E_F^- \cap C \subseteq E_F^+ \}
\]

\[
pref_C(F) = \{ E \in adm_C(F) \mid \text{for all } D \in adm_C(F) \text{ with } E \setminus C = D \setminus C, E_F^+ \setminus C \subseteq D_F^+ \setminus C, E_F^- \setminus E_F^+ \supseteq D_F^- \setminus D_F^+ : E \cap C \not\subset D \cap C \}
\]

For complete and grounded semantics, similar definitions can be given by using a parameterized version of the characteristic function.
Theoretical Foundations – Main Results

For other semantics, such variants can be defined accordingly.

**Definition**

Let $F$ be an AF, $C \subseteq U$. We define

\[
\text{adm}_C(F) = \{ E \in \text{cf}(F) \mid E_F^- \cap C \subseteq E_F^+ \} \\
\text{pref}_C(F) = \{ E \in \text{adm}_C(F) \mid \text{for all } D \in \text{adm}_C(F) \text{ with } E \setminus C = D \setminus C, E_F^+ \setminus C \subseteq D_F^+ \setminus C, E_F^- \setminus E_F^+ \supseteq D_F^- \setminus D_F^+ : E \cap C \not\subseteq D \cap C \} \]

For complete and grounded semantics, similar definitions can be given by using a parameterized version of the characteristic function.

**Theorem**

Let $F$ be an AF and $C \subseteq U$. Then, the following relations hold:

\[ stb_C(F) \subseteq \text{pref}_C(F) \subseteq \text{comp}_C(F) \subseteq \text{adm}_C(F); \text{grd}_C(F) \subseteq \text{comp}_C(F). \]
Theorem

Let $F, F'$ be AFs and $C \subseteq U$. Then, $F \equiv_C^{stb} F'$ iff jointly

1. $stb_C(F) = stb_C(F')$;
2. if $stb_C(F) \neq \emptyset$, $A(F) \setminus C = A(F') \setminus C$;
3. for all $E \in stb_C(F)$, $E_F^+ \setminus C = E_{F'}^+ \setminus C$. 
Theoretical Foundations – Main Results

**Theorem**

Let $F, F'$ be AFs and $C \subseteq U$. Then, $F \equiv_{stb}^C F'$ iff jointly

(1) $\text{stb}_C(F) = \text{stb}_C(F')$;

(2) if $\text{stb}_C(F) \neq \emptyset$, $A(F) \setminus C = A(F') \setminus C$;

(3) for all $E \in \text{stb}_C(F)$, $E_F^+ \setminus C = E_{F'}^+ \setminus C$.

**Example with** $C = \{c, d, e, f\}$

Recall (1) $\text{stb}_C(F) = \text{stb}_C(F') = \{\{a, d, e\}, \{b, c, e\}, \{d, e\}, \{c, e\}\}$; (2) and (3) also hold.
Theoretical Foundations – Main Results

**Theorem**

Let $F, F'$ be AFs and $C \subseteq U$. Then, $F \equiv_C^{stb} F'$ iff jointly

1. $stb_C(F) = stb_C(F')$;
2. if $stb_C(F) \neq \emptyset$, $A(F) \setminus C = A(F') \setminus C$;
3. for all $E \in stb_C(F)$, $E_F^+ \setminus C = E_{F'}^+ \setminus C$.

Similar characterization results can be shown for the other main semantics.
Replacement Theorem

For AFs $F, F', G$ and $C \subseteq U$ such that $A(F) \cup A(F') \subseteq C$, $(A(G) \setminus A(F)) \cap C = \emptyset$, and $F$ is a sub-AF of $G$, let $B = (A(F)) \oplus_G \cup (A(F)) \ominus_G$ and $F^G = (B, R(G) \cap (B \times B))$.

Then, $F^G \equiv^\sigma_C F^G[F/F']$ implies $G \equiv^\sigma G[F/F'].$
Replacement Theorem

For AFs $F, F', G$ and $C \subseteq U$ such that $A(F) \cup A(F') \subseteq C$, $(A(G) \setminus A(F)) \cap C = \emptyset$, and $F$ is a sub-AF of $G$, let $B = (A(F))_G \oplus (A(F))_G^-$ and $F^G = (B, R(G) \cap (B \times B))$. Then, $F^G \equiv^\sigma_C F^G[F/F']$ implies $G \equiv^\sigma G[F/F']$.

Example

G : \[
\begin{array}{c}
C_1 & \rightarrow & C_2 & \rightarrow & C_3 & \rightarrow & C_4 & \rightarrow & C_5 \\
C_{2n+1} & \leftarrow & a & \leftrightarrow & b & \rightarrow & c & \leftrightarrow & C_{2n} \\
C_{2n} & \rightarrow & \ldots & \rightarrow & C_6 & \rightarrow
\end{array}
\]

G[F/F'] : \[
\begin{array}{c}
C_1 & \rightarrow & C_2 & \rightarrow & C_4 & \rightarrow \\
C_{2n+1} & \leftarrow & a & \leftrightarrow & b & \rightarrow & c & \leftrightarrow & C_{2n} \\
C_{2n} & \rightarrow & \ldots & \rightarrow & C_6 & \rightarrow
\end{array}
\]
Some complexity results:

<table>
<thead>
<tr>
<th>$\sigma$</th>
<th>$\text{grd}$</th>
<th>$\text{stb}$</th>
<th>$\text{adm}$</th>
<th>$\text{comp}$</th>
<th>$\text{pref}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F \equiv_{S}^{\sigma} G$</td>
<td>in L</td>
<td>in L</td>
<td>in L</td>
<td>in L</td>
<td>in L</td>
</tr>
<tr>
<td>$F \equiv_{P}^{\sigma} G$</td>
<td>$P$-c</td>
<td>coNP-c.</td>
<td>coNP-c.</td>
<td>coNP-c.</td>
<td>$\Pi_{2}^{P}$-c.</td>
</tr>
<tr>
<td>$F \equiv_{C}^{\sigma} G$</td>
<td>coNP-c.</td>
<td>coNP-c.</td>
<td>coNP-c.</td>
<td>coNP-c.</td>
<td>$\Pi_{2}^{P}$-c.</td>
</tr>
</tbody>
</table>
Building a Preprocessing Machine - Our Vision
1. Collect patterns \((F^G, F, F')\) which apply for the replacement theorem

- This can be done in an offline-phase
- employ the equivalence characterizations
- different patterns for different semantics
2. Build a tool that scans a given AF for possible application of the replacement patterns \((F^G, F, F')\)

- Requires efficient implementation of subgraph-isomorphism problem
- Sort out which size of subgraphs allow for efficient scanning for patterns
- Integrate other known simplifications (computation of grounded extension) and interleave this with the applied replacements
3. Experimental Evaluation and Fine-Tuning

- which replacements actually help solvers?
  - Preprocessing on the argumentation level should go beyond preprocessing on encodings
- identification of “promising regions” (e.g. potential separation into SCCs)
- integration of ML techniques
Conclusion

- Increasing interest in development of AF solvers
- In other domains, preprocessing recognized as a crucial step to improve efficiency
- Nonmonotonic nature of argumentation semantics makes life complicated

In this talk:
- Introduced a suitable notion of equivalence to seek for simplification patterns
- Discussion of next steps towards practical realization of a preprocessing tool
  - Recall: this is beneficial for all solvers!
Future Work and Open Questions

- Understand $C$-relativized equivalence for further semantics
- What can be done for acceptance problems?
- Claim: preprocessing could be more powerful if we allow to shift from AFs to a more general formalism (for instance, SETAFs)
  - however, this requires solvers for this general formalism
Future Work and Open Questions

- Understand $C$-relativized equivalence for further semantics
- What can be done for acceptance problems?
- Claim: preprocessing could be more powerful if we allow to shift from AFs to a more general formalism (for instance, SETAFs)
  - however, this requires solvers for this general formalism

Thank you for your attention!
