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

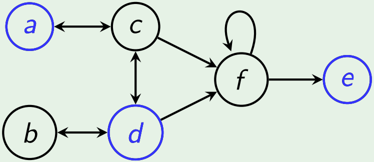
Prologue

Seminal Paper by Phan Minh Dung:
On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell., 77(2):321–358, 1995.

Prologue

Seminal Paper by Phan Minh Dung:
On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell., 77(2):321–358, 1995.

Example

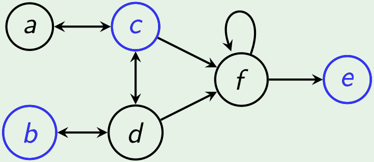


$stb(F) \quad = \{\{a, d, e\},$

Prologue

Seminal Paper by Phan Minh Dung:
On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell., 77(2):321–358, 1995.

Example

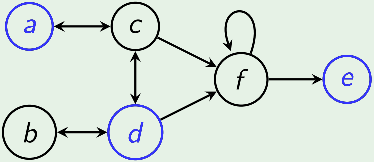


$stb(F) = \{\{a, d, e\}, \{b, c, e\}\}$

Prologue

Seminal Paper by Phan Minh Dung:
On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell., 77(2):321–358, 1995.

Example

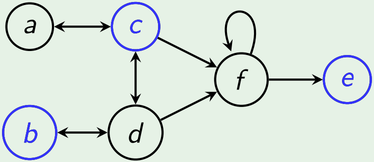


$$stb(F) = \{\{a, d, e\}, \{b, c, e\}\}$$
$$pref(F) = \{\{a, d, e\},$$

Prologue

Seminal Paper by Phan Minh Dung:
On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell., 77(2):321–358, 1995.

Example

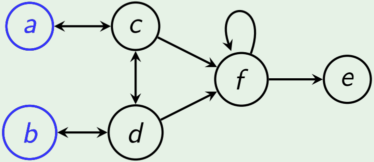


$$
\begin{aligned}
stb(F) &= \{\{a, d, e\}, \{b, c, e\}\} \\
pref(F) &= \{\{a, d, e\}, \{b, c, e\},
\end{aligned}
$$

Prologue

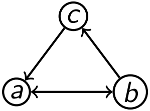
Seminal Paper by Phan Minh Dung:
On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell., 77(2):321–358, 1995.

Example



$$
\begin{aligned}
stb(F) &= \{\{a, d, e\}, \{b, c, e\}\} \\
pref(F) &= \{\{a, d, e\}, \{b, c, e\}, \{a, b\}\}
\end{aligned}
$$

Prologue



Nonmon. Reasoning

Logic Programming

n-Person Games

⋮

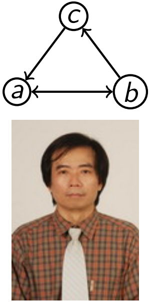
ABA

ASPIC

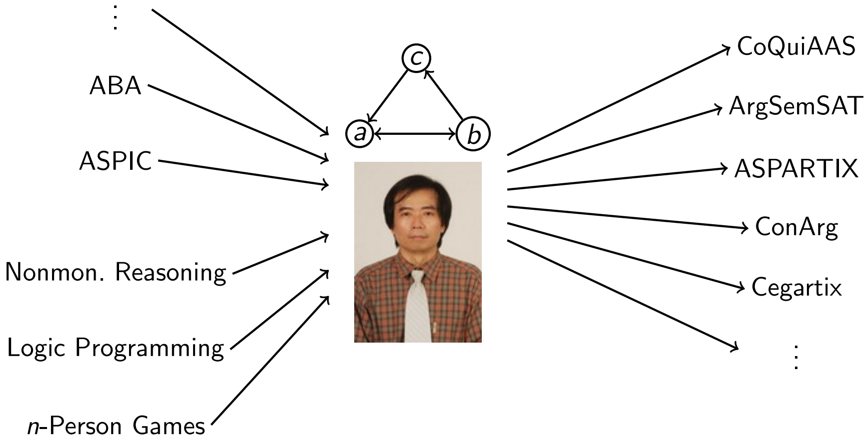
Nonmon. Reasoning

Logic Programming

n-Person Games

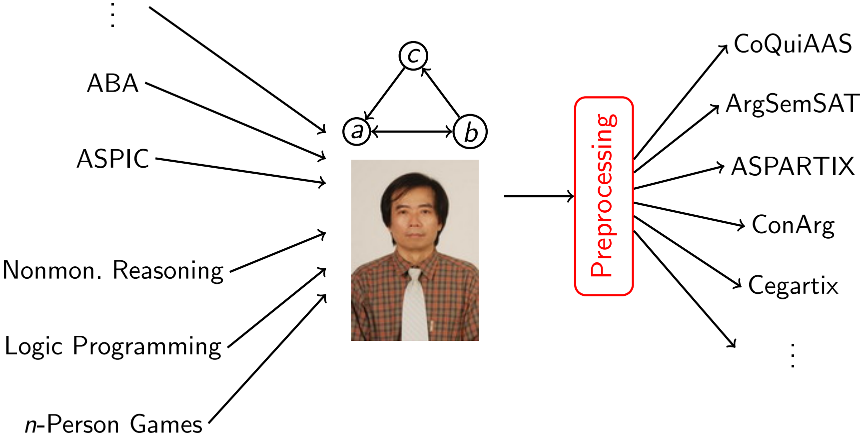


Prologue



⋮
ABA
ASPIC
Nonmon. Reasoning
Logic Programming
n-Person Games
c
a
b
CoQuiAAS
ArgSemSAT
ASPARTIX
ConArg
Cegartix
⋮

Prologue


⋮
ABA
ASPIC
Nonmon. Reasoning
Logic Programming
n-Person Games
c
a
b
Preprocessing
CoQuiAAS
ArgSemSAT
ASPARTIX
ConArg
Cegartix
⋮

Outline

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

Background

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

Background

σ	$Cred_\sigma$	$Skept_\sigma$	Ver_σ	NE_σ
cf	in L	trivial	in L	in L
grd	P-c	P-c	P-c	in L
stb	NP-c	coNP-c	in L	NP-c
adm	NP-c	trivial	in L	NP-c
comp	NP-c	P-c	in L	NP-c
ideal	in Θ_2^P	in Θ_2^P	in Θ_2^P	in Θ_2^P
pref	NP-c	Π_2^P-c	coNP-c	NP-c
sem	Σ_2^P-c	Π_2^P-c	coNP-c	NP-c
stage	Σ_2^P-c	Π_2^P-c	coNP-c	in L

Background - ICCMA'17

(http://www.dbai.tuwien.ac.at/iccma17/)

Home | Calls | Rules | Participation | Submissions | Results | Organization

ICCMA 2017

The tasks supported by the solvers are summarized in the following table:

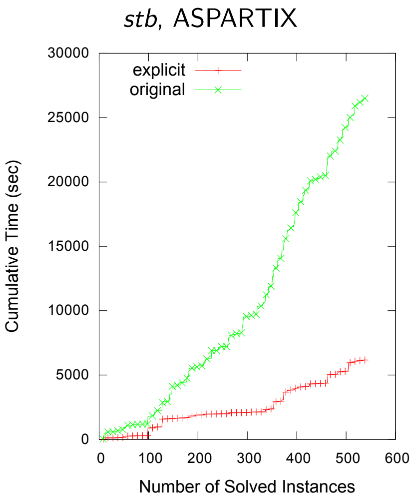
	D3	CO				PR				ST				SST				STG				GR		ID	
		DC	DS	SE	EE	DC	DS	SE	EE	DC	DS	SE	EE	DC	DS	SE	EE	DC	DS	SE	EE	DC	SE	DC	SE
argmat olpb		1	1	1	1					1	1	1	1									1	1		
argmat dvisat	1	1	1	1	1	1	1	1	1	1	1	1	1									1	1	1	1
argmat-mpg	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
argmat-sat	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ArgSemSAT		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1					1	1		
ArgTools		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
ASPrMin									1																
cegartix	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Chimaerarg									1				1												
ConArg	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
CoQuiAAS	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
EqArgSolver	1	1	1	1	1	1	1	1	1	1	1	1	1									1	1		
gg-sts	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
goDIAMOND	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
heureka		1	1	1	1	1	1	1	1	1	1	1	1									1	1		
pyglaf	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

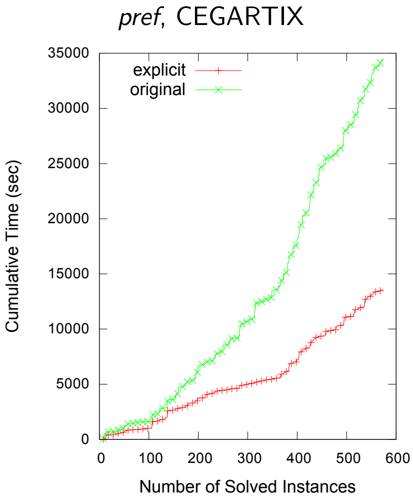
Preprocessing

- 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 Bloqqer 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.

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

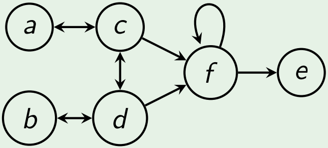
Definition

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

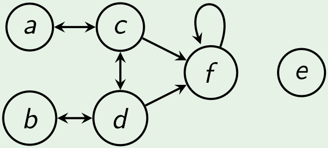
Definition

Given a semantics σ. Two AFs F and F' are strongly equivalent w.r.t. σ (in symbols $F \equiv^{\sigma}_{\mathcal{S}} F'$) iff $F \cup H \equiv^{\sigma} F' \cup H$ holds for each AF H.

Example



$\equiv^{stb}$

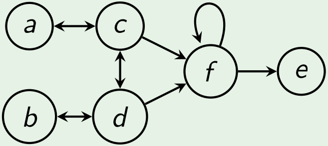


$stb(F) = \{\{a, d, e\}, \{b, c, e\}\}$

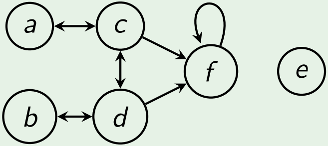
$stb(F') = \{\{a, d, e\}, \{b, c, e\}\}$

Theoretical Foundations – Notions of Equivalence

Example



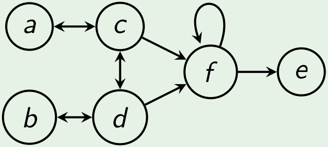
$$\equiv_{\mathcal{S}}^{stb}$$



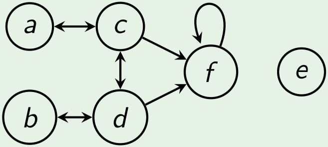
Follows from results in [Oikarinen & Woltran, 2011].

Theoretical Foundations – Notions of Equivalence

Example



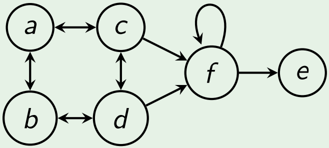
$\not\equiv^{pref}$



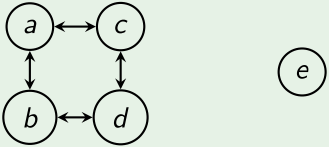
$pref(F) =$
$\{\{a, d, e\}, \{b, c, e\}, \{a, b\}\}$

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

Example



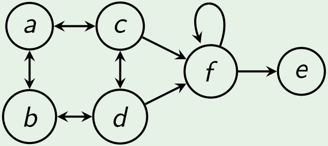
$\equiv^{stb}$



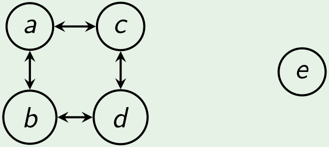
$stb(F) = \{\{a, d, e\}, \{b, c, e\}\}$

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

Example

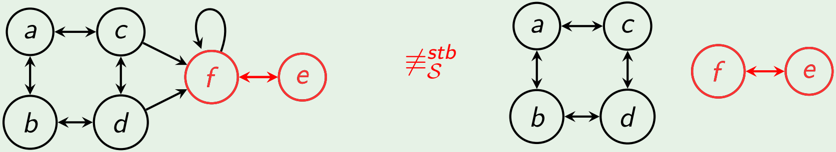


$$\not\equiv^{stb}_{\mathcal{S}}$$



Theoretical Foundations – Notions of Equivalence

Example



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

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': $F \equiv^{\sigma}_{\mathcal{S}} F'$ 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': $F \equiv^{\sigma}_{\mathcal{S}} F'$ iff $F = F'$!

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

Definition

Given a semantics σ and arguments $C \subseteq U$. Two AFs F and F' are C-relativized equivalent w.r.t. σ (in symbols $F \equiv^{\sigma}_{C} F'$) 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': $F \equiv^{\sigma}_{\mathcal{S}} F'$ iff $F = F'$!

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

Definition

Given a semantics σ and arguments $C \subseteq U$. Two AFs F and F' are *C-relativized equivalent* w.r.t. σ (in symbols $F \equiv^{\sigma}_{C} F'$) 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

Theoretical Foundations – Main Results

Definition

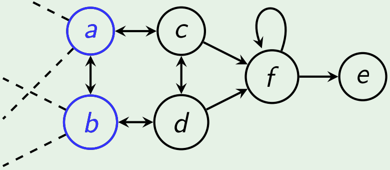
Given a semantics σ and arguments $C \subseteq U$. Two AFs F and F' are C-relativized equivalent w.r.t. σ (in symbols $F \equiv^{\sigma}_{C} F'$) iff $F \cup H \equiv^{\sigma} F' \cup H$ holds for each AF H not containing arguments from C.

Theoretical Foundations – Main Results

Definition

Given a semantics σ and arguments $C \subseteq U$. Two AFs F and F' are C-relativized equivalent w.r.t. σ (in symbols $F \equiv^{\sigma}_{C} F'$) iff $F \cup H \equiv^{\sigma} F' \cup H$ 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

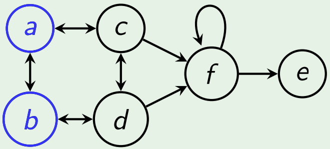
We first define a parameterized notion of the semantics.

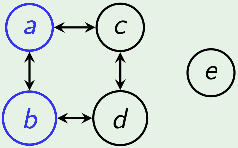
Definition

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

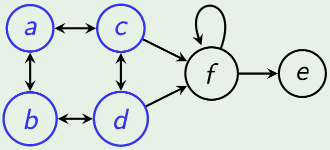
We first define a parameterized notion of the semantics.

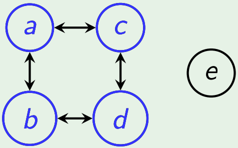
Definition

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

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

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





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

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

Theoretical Foundations – Main Results

For other semantics, such variants can be defined accordingly.

Definition

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

$$
\begin{aligned}
adm_C(F) &= \{E \in cf(F) \mid E_F^- \cap C \subseteq E_F^+\} \\
pref_C(F) &= \{E \in adm_C(F) \mid \text{ for all } D \in adm_C(F) \text{ with} \\
&\quad E \setminus C = D \setminus C, E_F^+ \setminus C \subseteq D_F^+ \setminus, E_F^- \setminus E_F^+ \supseteq D_F^- \setminus D_F^+ : \\
&\quad E \cap C \not\subset D \cap C\}
\end{aligned}
$$

Theoretical Foundations – Main Results

For other semantics, such variants can be defined accordingly.

Definition

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

$$
\begin{aligned}
adm_C(F) &= \{E \in cf(F) \mid E_F^- \cap C \subseteq E_F^+\} \\
pref_C(F) &= \{E \in adm_C(F) \mid \text{ for all } D \in adm_C(F) \text{ with} \\
&\quad E \setminus C = D \setminus C, E_F^+ \setminus C \subseteq D_F^+ \setminus, E_F^- \setminus E_F^+ \supseteq D_F^- \setminus D_F^+ : \\
&\quad E \cap C \not\subset D \cap C\}
\end{aligned}
$$

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

$$\begin{aligned} adm_C(F) &= \{E \in cf(F) \mid E^-_F \cap C \subseteq E^+_F\} \\ pref_C(F) &= \{E \in adm_C(F) \mid \text{ for all } D \in adm_C(F) \text{ with} \\ &\quad E \setminus C = D \setminus C, E^+_F \setminus C \subseteq D^+_F \setminus, E^-_F \setminus E^+_F \supseteq D^-_F \setminus D^+_F : \\ &\quad E \cap C \not\subset D \cap C\} \end{aligned}$$

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:
$stb_C(F) \subseteq pref_C(F) \subseteq comp_C(F) \subseteq adm_C(F)$; $grd_C(F) \subseteq comp_C(F)$.

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* $E \in stb_C(F)$, $E_F^+ \setminus C = E_{F'}^+ \setminus C$.

Theoretical Foundations – Main Results

Theorem

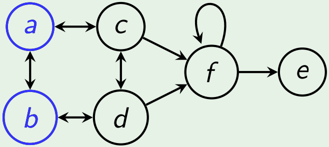
Let F, F' be AFs and $C \subseteq U$. Then, $F \equiv^{stb}_{C} F'$ iff jointly

(1) $stb_C(F) = stb_C(F')$;

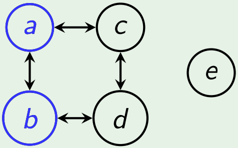
(2) *if* $stb_C(F) \neq \emptyset$, $A(F) \setminus C = A(F') \setminus C$;

(3) *for all* $E \in stb_C(F)$, $E^+_F \setminus C = E^+_{F'} \setminus C$.

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



$\equiv^{stb}_{C}$



Recall (1) $stb_C(F) = stb_C(F') = \{\{a, d, e\}, \{b, c, e\}, \{d, e\}, \{c, e\}\}$;
(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.

Theoretical Foundations – Main Results

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 $B = (A(F))^{\oplus}_{G} \cup (A(F))^{-}_{G}$ and $F^G = (B, R(G) \cap (B \times B))$.

Then, $F^G \equiv^{\sigma}_{C} F^G[F/F']$ implies $G \equiv^{\sigma} G[F/F']$.

Theoretical Foundations – Main Results

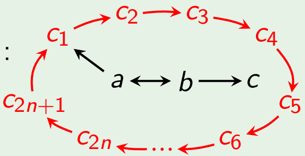
Replacement Theorem

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 $B = (A(F))^{\oplus}_{G} \cup (A(F))^{-}_{G}$ and $F^G = (B, R(G) \cap (B \times B))$.

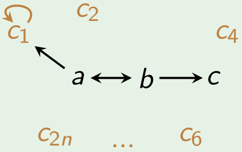
Then, $F^G \equiv^{\sigma}_{C} F^G[F/F']$ implies $G \equiv^{\sigma} G[F/F']$.

Example

G :



$G[F/F']$:

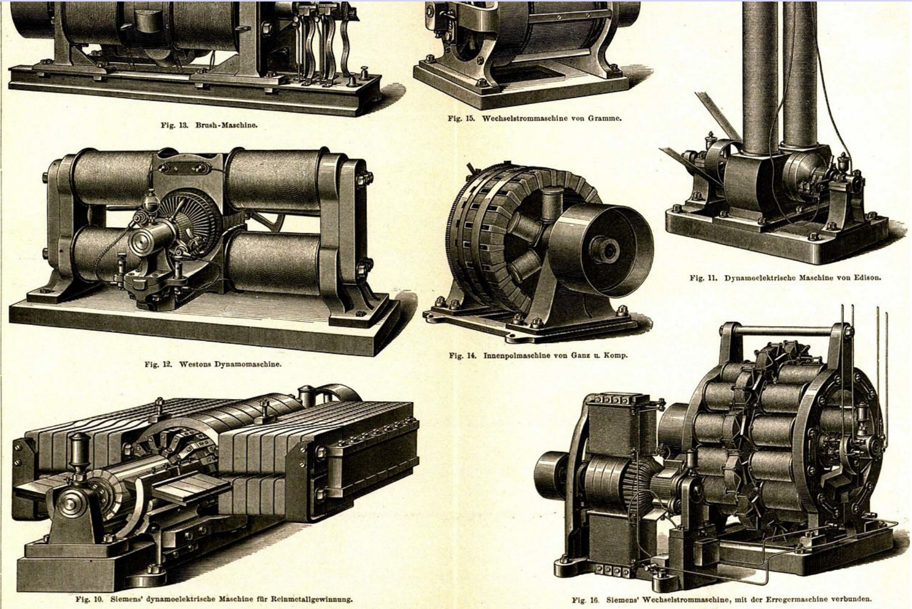


Theoretical Foundations – Main Results

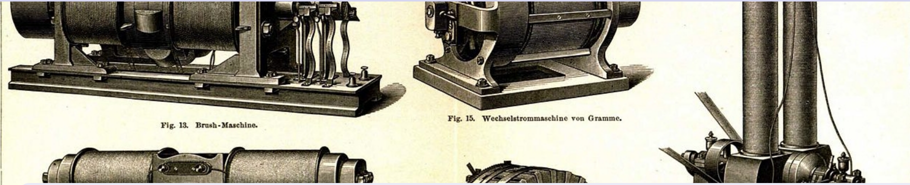
Some complexity results:

σ	*grd*	*stb*	*adm*	*comp*	*pref*
$F \equiv^{\sigma}_{\mathcal{S}} G$	in L	in L	in L	in L	in L
$F \equiv^{\sigma} G$	P-c	coNP-c.	coNP-c.	coNP-c.	Π^P_2-c.
$F \equiv^{\sigma}_{C} G$	coNP-c.	coNP-c.	coNP-c.	coNP-c.	Π^P_2-c.

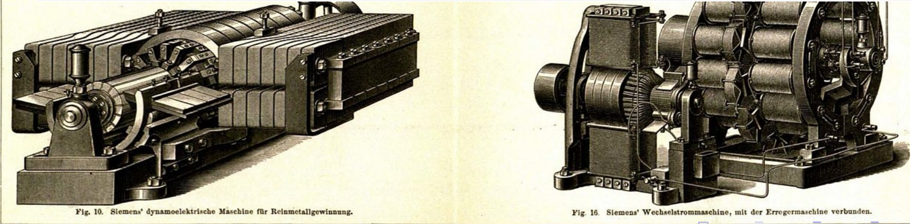
Building a Preprocessing Machine - Our Vision



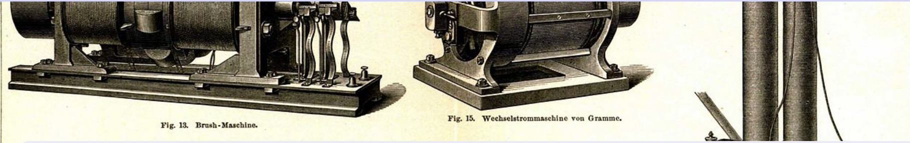
Building a Preprocessing Machine - Our Vision



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

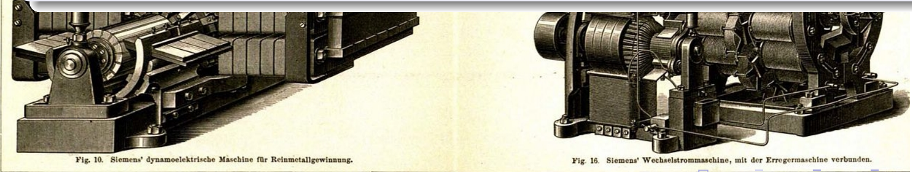


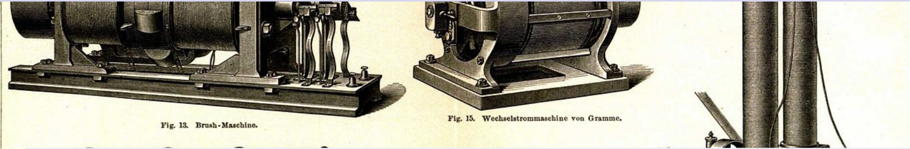
Building a Preprocessing Machine - Our Vision



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!

References

- P. M. Dung: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell. 77(2):321–358, 1995.
- R. Baumann, W. Dvořák, T. Linsbichler and S. Woltran: A General Notion of Equivalence for Abstract Argumentation. Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17), pages 800–806. AAAI Press, 2017. Full version: `www.dbai.tuwien.ac.at/research/report/dbai-tr-2017-105.pdf`
- E. Oikarinen, S. Woltran: Characterizing strong equivalence for argumentation frameworks. Artif. Intell. 175(14-15):1985–2009, 2011.