Abstract Dialectical Frameworks*
Properties, Complexity, and Implementation

Stefan Ellmauthaler

Intelligent Systems
Computer Science Institute
University of Leipzig

15.11.2012

* The presented results are from the same-titled Master's thesis, done at the Vienna University of Technology (Institute of Information Systems, Database and Artificial Intelligence Group)
Motivation - Argumentation

- Situated in the intersection between
  - Philosophy,
  - Artificial Intelligence, and
  - several application domains.

- Formal approach to **nonmonotonic reasoning** with potentially inconsistent knowledge
Motivation - Argumentation

Situated in the intersection between

- Philosophy,
- Artificial Intelligence, and
- several application domains.

Formal approach to nonmonotonic reasoning with potentially inconsistent knowledge

Concerns of Argumentation Models

- representation of **arguments**
- representation of **relations** between arguments
- finding “acceptable” **sets of arguments** with **semantics**
  - acceptable set is an extension
  - arguments are **defeasible** during resolving of extensions
Motivation - ADFs

- **Dung’s Argumentation Framework**
  - introduced by [Dung, 1995]
  - simple
  - powerful
- Dung’s AF can only model attack relations natively
- More complex relations need auxiliary constructs
**Motivation - ADFs**

- **Dung’s Argumentation Framework**
  - introduced by [Dung, 1995]
  - simple
  - powerful

- Dung’s AF can only model attack relations natively

- More complex relations need auxiliary constructs
Motivation - ADFs

- Dung’s Argumentation Framework
  - introduced by [Dung, 1995]
  - simple
  - powerful
- Dung’s AF can only model attack relations natively
- More complex relations need auxiliary constructs

- Abstract Dialectical Frameworks
  - introduced by [Brewka and Woltran, 2010]
  - generalization of Dung’s AF
  - total functions specify relation types (acceptance conditions)
  - bipolar Abstract Dialectical Frameworks (BADFs) restrict relation types to be attacking or supporting
  - some semantics are only defined for BADFs
Main contributions

- **Alternative representations** for ADFs with useful properties
- **Generalized** and unrestricted **stable model semantics** for ADFs
- **Implementation** of a software system to compute the extensions under several semantics
Propositional Formula ADF

Definition (pForm-ADF)
A pForm-ADF is a pair $D = (S, AC)$, where
- $S$ is a set of statements
- $AC = \{AC_s\}_{s \in S}$ is the set of acceptance conditions, where each statement has exactly one associated condition.

An acceptance condition $AC_s$ is a propositional formula $\psi$. 
Definition (pForm-ADF)

A pForm-ADF is a pair $D = (S, AC)$, where

- $S$ is a set of statements
- $AC = \{AC_s\}_{s \in S}$ is the set of acceptance conditions, where each statement has exactly one associated condition.

An acceptance condition $AC_s$ is a propositional formula $\psi$.

Definition (model semantics)

Let $D = (S, AC)$ be a pForm-ADF. $M \subseteq S$ is a model of $D$ if for each $s \in S$, $M \in mod_p(AC_s)$ iff $s \in M$, holds. $model_{pADF}(D)$ is the set of models for the pForm-ADF $D$. 
Propositional Formula ADF

Example (pForm-ADF)

\[ S = \{A, B, C\} \]
\[ AC = \{AC_A, AC_B, AC_C\} \]
\[ AC_A = B \]
\[ AC_B = A \]
\[ AC_C = \neg B \]
\[ \text{models} = \{\{A, B\}, \{C\}\} \]
Propositional Formula ADF

Example (pForm-ADF)

\[ S = \{A, B, C, D\} \]
\[ AC = \{AC_A, AC_B, AC_C, AC_D\} \]
\[ AC_A = \top \]
\[ AC_B = \neg A \]
\[ AC_C = A \]
\[ AC_D = (\neg B \land C) \lor (B \land \neg A) \]
\[ \text{models} = \\{\{A, C, D\}\} \]
Stable model semantics

It is based on the transformation from an ADF to a BADF:

- splits acceptance conditions with dependent links
- one AC with supporting character
- one AC with attacking character
- done by additional criteria in the ACs

Example:

\[ \text{AC}\_s = s' \lor s'' \]
\[ \text{AC}\_s' = (a \land b) \lor (\neg a \land c) \]
\[ \text{AC}\_s'' = (a \land b) \lor (\neg a \land c) \]
Stable model semantics

It is based on the transformation from an ADF to a BADF:
- splits acceptance conditions with dependent links
- one AC with supporting character
- one AC with attacking character
- done by additional criteria in the ACs

Example

\[ AC_s = (a \land b) \lor (\neg a \land c) \]
Stable model semantics

It is based on the transformation from an ADF to a BADF:

- splits acceptance conditions with dependent links
- one AC with supporting character
- one AC with attacking character
- done by additional criteria in the ACs

Example

\[ AC_s = (a \land b) \lor (\neg a \land c) \]
\[ AC_{s'} = ((a \land b) \lor (\neg a \land c)) \]
\[ AC_{s''} = ((a \land b) \lor (\neg a \land c)) \]
Stable model semantics

It is based on the transformation from an ADF to a BADF:

- splits acceptance conditions with dependent links
- one AC with supporting character
- one AC with attacking character
- done by additional criteria in the ACs

Example

\[
AC_s = s' \lor s'' \\
AC_{s'} = ((a \land b) \lor (\neg a \land c)) \\
AC_{s''} = ((a \land b) \lor (\neg a \land c))
\]
Stable model semantics

It is based on the transformation from an ADF to a BADF:

- splits acceptance conditions with dependent links
- one AC with supporting character
- one AC with attacking character
- done by additional criteria in the ACs

Example

\[ AC_s = s' \lor s'' \]
\[ AC_{s'} = ((a \land b) \lor (\neg a \land c)) \land a \]
\[ AC_{s''} = ((a \land b) \lor (\neg a \land c)) \land \neg a \]
Stable model semantics

- stable semantics for bipolar pForm-ADFs
- generalization lifts the restriction of bipolar ADFs
Stable model semantics

- stable semantics for bipolar pForm-ADFs
- generalization lifts the restriction of bipolar ADFs

**Definition ((generalized) stable model for pForm-ADFs)**

Let \( D = (S, AC) \) be a (bipolar) pForm-ADF. A model \( M \) of \( D \) is a stable model if \( M \) is the least model of the reduced pForm-ADF \( D^M = (S^M, AC^M) \) obtained from \( D \) by

(I) eliminating all nodes not contained in \( M \), s.t. \( S^M = S \cap M \),

(II) for all \( s \in S^M \) substitute in \( AC_s \) all \( a \in \text{atoms}(AC_s) \) with \( \bot \) if \( a \notin S^M \),

(III) for all \( s \in S^M \) substitute in \( AC_s \) all \( a \in \text{atoms}(AC_s) \) with \( \bot \) if \( a \in \text{att}(AC_s) \).
Stable model semantics

- stable semantics for bipolar pForm-ADFs
- generalization lifts the restriction of bipolar ADFs

**Definition ((generalized) stable model for pForm-ADFs)**

Let $D = (S, AC)$ be a (bipolar) pForm-ADF. A model $M$ of $D$ is a stable model if $M$ is the least model of the reduced pForm-ADF $D^M = (S^M, AC^M)$ obtained from $D$ by

1. eliminating all nodes not contained in $M$, s.t. $S^M = S \cap M$,
2. for all $s \in S^M$ substitute in $AC_s$ all $a \in \text{atoms}(AC_s)$ with $\bot$ if $a \not\in S^M$,
3. for all $s \in S^M$ substitute in $AC_s$ all $a \in \text{atoms}(AC_s)$ with $\bot$ if $a \in \text{att}(AC_s)$.
4. for all $s \in S^M$, if $\{a_1, \ldots, a_n\}$ is the set of all selected dependent variables in $AC_s$ and $M$ then $AC^M_s = AC_s \land a_1 \land \ldots \land a_n$. 

Stefan Ellmuthaler, 15.11.2012
Abstract Dialectical Frameworks
ASP encoding

- Encoding for all semantics [Ellmauthaler and Wallner, 2012]
- Based on pForm-ADF representation
- Utilize different logic programming techniques
  - Guess & Check
  - Saturation
  - Optimization
  - Subset-maximality
  - Iterations
- Implementation uses the Potassco Answer Set Solving Collection [Gebser et al., 2011]
Example (Instance format)

- `statement(a)`.  
- `ac(a, b)`.  
- `supp(b, a)`.  
- `statement(b)`.  
- `ac(b, a)`.  
- `supp(a, b)`.  
- `statement(c)`.  
- `ac(c, neg(b))`.  
- `att(b, c)`.

Example

```
sup A
att B
sup

att
C
```
ASP Encoding

Example (Instance format)

\begin{align*}
\text{statement}(a). & \quad \text{ac}(a,b). & \quad \text{supp}(b,a). \\
\text{statement}(b). & \quad \text{ac}(b,a). & \quad \text{supp}(a,b). \\
\text{statement}(c). & \quad \text{ac}(c,\neg(b)). & \quad \text{att}(b,c). \\
\end{align*}

Example

\begin{align*}
\text{in}(X) & :\neg \text{out}(X), \text{statement}(X). \\
\text{out}(X) & :\neg \text{in}(X), \text{statement}(X). \\
\text{nomodel}(F) & :\neg \text{in}(X), \text{ac}(X,F). \\
\text{ismodel}(F) & :\neg \text{out}(X), \text{ac}(X,F). \\
\end{align*}
Implementation for the following semantics

- conflict-free set
- model
- linktype distinction
- stable model
- admissible set
- preferred model
- well-founded model

Preliminary benchmark tests for BADFs with up to 30 statements and up to 8 links per statement
Alternative Representations for ADFs
- Propositional Formula ADFs
- Hypergraph ADFs

Subclass for BADFs on pForm-ADFs (monotone pForm-ADF)

ADF → BADF transformation

Unrestricted generalized stable models semantics

Complexity results for link-type decision problem for ADFs (coNP-complete)

Complexity results for the generalized stable model semantics ($CA^{monotone} = NP$-complete)

Counter-examples where AF based inter-semantics relations for ADFs do not hold
Many **different approaches** based on Dung’s AF, like

- Constraint Argumentation Frameworks (CAF) [Coste-Marquis et al., 2006],
- Extended Argumentation Frameworks (EAF) [Modgil, 2009],
- Argumentation Frameworks with Recursive Attacks (AFRA) [Baroni et al., 2011],
- Context Based Argumentation [Brewka and Eiter, 2009], and
- Managed Multi Context Systems (mMCS) [Brewka et al., 2011].

**Carneades** [Gordon et al., 2007]

- is used for law interpretation
- utilizes another approach
- multiple stages of computation
- one fixed stage can be simulated with ADFs [Brewka and Gordon, 2010]
Future Work

- **Further investigations** of inter-semantic relations and possibly revamping some semantics
- **Further investigation** of the correspondence between stable model semantics and the Gelfond-Lifschitz reduct for Logic Programming
- **Simulations** of CAF, EAF, AFRA, ... with ADFs
- **Enhance** mMCS with ADFs
- **Optimization** of the implementation
- **Utilization** of other argumentation systems for AFs (e.g. CEGARTIX, DYNPARTIX)


