A Comprehensive Analysis of the cf2 Argumentation Semantics: From Characterization to Implementation

Sarah Alice Gaggl

Institute of Informationsystems, Vienna University of Technology

Vienna — March 4, 2013
Argumentation is one of the major fields in Artificial Intelligence (AI).

Applications in diverse domains (legal reasoning, multi-agent systems, social networks, e-government, decision support).

Concept of abstract Argumentation Frameworks (AFs) [Dung, 1995] is one of the most popular approaches.

Arguments and a binary attack relation between them, denoting conflicts, are the only components.

Numerous semantics to solve the inherent conflicts by selecting acceptable sets of argument.

Admissible-based versus naive-based semantics.

Development of competitive systems.
Motivation ctd.

cf² Semantics

- is based on decomposition of the framework along its strongly connected components (SCCs) [Baroni et al., 2005];
- does not require to defend arguments against attacks;
- allows to treat cycles in a more sensitive way than other semantics;
- is not well studied, due to quite complicated definition.
Motivation ctd.

cf2 Semantics

- is based on decomposition of the framework along its strongly connected components (SCCs) [Baroni et al., 2005];
- does not require to defend arguments against attacks;
- allows to treat cycles in a more sensitive way than other semantics;
- is not well studied, due to quite complicated definition.

Goals of the Thesis

- Answer-set programming encodings for cf2.
- Alternative characterization.
- Verification of behavior on concrete instances.
- Identification of possible redundancies.
- Complete complexity analysis.
1. Background on abstract argumentation frameworks and semantics
2. Alternative characterization of cf2
3. Combining cf2 and stage semantics
4. Redundancies and strong equivalence
5. Computational complexity
6. Implementations
7. Conclusion
Argumentation Framework

Abstract Argumentation Framework [Dung, 1995]

An abstract argumentation framework \((AF)\) is a pair \(F = (A, R)\), where \(A\) is a finite set of arguments and \(R \subseteq A \times A\). Then \((a, b) \in R\) if \(a\) attacks \(b\). Argument \(a \in A\) is defended by \(S \subseteq A\) (in \(F\)) iff, for each \(b \in A\) with \((b, a) \in R\), \(S\) attacks \(b\).

Example
Semantics for AFs

Let $F = (A, R)$ and $S \subseteq A$, we say

- $S$ is conflict-free in $F$, i.e. $S \in cf(F)$, if $\forall a, b \in S: (a, b) \notin R$;
- $S \in cf(F)$ is maximal conflict-free or naive (in $F$), i.e. $S \in naive(F)$, if $\forall T \in cf(F), S \not\subset T$.

Example

$naive(F) = \{\{a, d, g\}, \{a, c, e\}, \{a, c, g\}\}$. 
Naive-based Semantics

Let $F = (A, R)$ and $S \subseteq A$. Let $S^+_R = S \cup \{b \mid \exists a \in S, \text{ s. t. } (a, b) \in R\}$ be the range of $S$. Then, a set $S \in cf(F)$ is

- a stable extension (of $F$), i.e. $S \in \text{stable}(F)$, if $S^+_R = A$;
- stage in $F$, i.e. $S \in \text{stage}(F)$, if for each $T \in cf(F)$, $S^+_R \not\subseteq T^+_R$.

Example

$$\text{stable}(F) = \emptyset, \text{stage}(F) = \{\{a, d, g\}, \{a, c, e\}, \{a, c, g\}\}.$$
Admissible-based Semantics

Then, \( S \in cf(F) \) is

- admissible in \( F \), i.e. \( S \in adm(F) \), if each \( a \in S \) is defended by \( S \);
- a preferred extension (of \( F \)), i.e. \( S \in pref(F) \), if \( S \in adm(F) \) and for each \( T \in adm(F) \), \( S \not\subset T \).

Example

\[
adm(F) = \{\emptyset, \{a\}, \{c\}, \{d\}, \{a, c\}, \{a, d\}\}, \quad pref(F) = \{\{a, c\}, \{a, d\}\}.
\]
One of the SCC-recursive semantics introduced in [Baroni et al., 2005].

Naive-based semantics.

Handles odd- and even-length cycles in a uniform way.

Can accept arguments out of odd-length cycles.

Can accept arguments attacked by self-attacking arguments.

Satisfies most of the evaluation criteria proposed in [Baroni and Giacomin, 2007].
One of the SCC-recursive semantics introduced in [Baroni et al., 2005].

Naive-based semantics. Handles odd- and even-length cycles in a uniform way.

Can accept arguments out of odd-length cycles. Can accept arguments attacked by self-attacking arguments.

Satisfies most of the evaluation criteria proposed in [Baroni and Giacomin, 2007].

Further Notations, let $F = (A, R)$

- $SCCs(F)$: set of strongly connected components of $F$,
- $C_F(a)$: the unique set $C \in SCCs(F)$, s.t. $a \in C$,
- $F|_S = ((A \cap S), R \cap (S \times S))$: sub-framework of $F$ w.r.t. $S$,
- $F|_S - S' = F|_{S \setminus S'}, F - S = F|_{A \setminus S}$. 
Definition \((D_F(S))\)

Let \(F = (A, R)\) be an AF and \(S \subseteq A\). An argument \(b \in A\) is component-defeated by \(S\) (in \(F\)), if there exists an \(a \in S\), such that \((a, b) \in R\) and \(a \notin C_F(b)\). The set of arguments component-defeated by \(S\) in \(F\) is denoted by \(D_F(S)\).
Definition \( (D_F(S)) \)

Let \( F = (A, R) \) be an AF and \( S \subseteq A \). An argument \( b \in A \) is **component-defeated** by \( S \) (in \( F \)), if there exists an \( a \in S \), such that \( (a, b) \in R \) and \( a \notin C_F(b) \). The set of arguments component-defeated by \( S \) in \( F \) is denoted by \( D_F(S) \).

**cf2 Extensions [Baroni et al., 2005]**

Let \( F = (A, R) \) be an argumentation framework and \( S \) a set of arguments. Then, \( S \) is a **cf2 extension** of \( F \), i.e. \( S \in cf2(F) \), iff

- \( S \in naive(F) \), in case \( |SCCs(F)| = 1 \);
- otherwise, \( \forall C \in SCCs(F), (S \cap C) \in cf2(F|_C - D_F(S)) \).
$S \in cf2(F)$ iff,

- $S \in naive(F)$, in case $|SCCs(F)| = 1$;
- otherwise, $\forall C \in SCCs(F), (S \cap C) \in cf2(F|_{C - D_F(S)})$.

Example

$S = \{a, d, e, g, i\}$, $S \in cf2(F)$?
$S \in \text{cf2}(F)$ iff,

- $S \in \text{naive}(F)$, in case $|\text{SCCs}(F)| = 1$;
- otherwise, $\forall C \in \text{SCCs}(F), (S \cap C) \in \text{cf2}(F|_{C - \text{DF}(S)})$.

**Example**

$S = \{a, d, e, g, i\}, S \in \text{cf2}(F)\,?\quad C_1 = \{a, b, c\}, C_2 = \{d\}, C_3 = \{e, f, g, h, i\}$ and $\text{DF}(S) = \{f\}$.
$S \in \text{cf2}(F)$ iff,

- $S \in \text{naive}(F)$, in case $|\text{SCCs}(F)| = 1$;
- otherwise, $\forall C \in \text{SCCs}(F)$, $(S \cap C) \in \text{cf2}(F|_C - D_F(S))$.

Example

$S = \{a, d, e, g, i\}, S \in \text{cf2}(F)$? $C_1 = \{a, b, c\}, C_2 = \{d\}, C_3 = \{e, f, g, h, i\}$ and $D_F(S) = \{f\}$.
$S \in cf2(F)$ iff,

- $S \in naive(F)$, in case $|SCCs(F)| = 1$;
- otherwise, $\forall C \in SCCs(F)$, $(S \cap C) \in cf2(F|_C - D_F(S))$.

**Example**

$S = \{a, d, e, g, i\}$, $S \in cf2(F)$? $C_4 = \{e\}$, $C_5 = \{g\}$, $C_6 = \{h\}$, $C_7 = \{i\}$ and $D_F|_{\{e,g,h,i\}}(\{e, g, i\}) = \{h\}$. 
Original definition of $cf_2$ is rather cumbersome to be directly encoded in ASP due to the recursive computation of different sub-frameworks.

In alternative characterization we shift the recursion to a certain set of arguments.

This enables to directly

- guess a set $S$;
- check whether $S$ is a naive extension of a certain instance of $F$. 
Separation

An AF $F = (A, R)$ is called separated if for each $(a, b) \in R$, there exists a path from $b$ to $a$. We define $[[F]] = \bigcup_{C \in SCCs(F)} F|_C$ and call $[[F]]$ the separation of $F$.

Example
Separation

An AF $F = (A, R)$ is called separated if for each $(a, b) \in R$, there exists a path from $b$ to $a$. We define $[[F]] = \bigcup_{C \in SCCs(F)} F|_C$ and call $[[F]]$ the separation of $F$.

Example
Reachability

Let $F = (A, R)$ be an AF, $B$ a set of arguments, and $a, b \in A$. We say that $b$ is reachable in $F$ from $a$ modulo $B$, in symbols $a \Rightarrow^B_F b$, if there exists a path from $a$ to $b$ in $F|_B$. 
Reachability

Let $F = (A, R)$ be an AF, $B$ a set of arguments, and $a, b \in A$. We say that $b$ is reachable in $F$ from $a$ modulo $B$, in symbols $a \Rightarrow_B^F b$, if there exists a path from $a$ to $b$ in $F|_B$.

Definition ($\Delta_{F,S}$)

For an AF $F = (A, R)$, $D \subseteq A$, and a set $S$ of arguments,

$$\Delta_{F,S}(D) = \{ a \in A \mid \exists b \in S : b \neq a, (b, a) \in R, a \not\Rightarrow_{F\setminus D} b \},$$

and $\Delta_{F,S}$ be the least fixed-point of $\Delta_{F,S}(\emptyset)$.
cf2 Extensions [Gaggl and Woltran, 2012]

Given an AF $F = (A, R)$.

$$\text{cf2}(F) = \{ S \mid S \in \text{naive}(F) \cap \text{naive}([F - \Delta_{F,S}]) \}.$$
\textit{cf2} Extensions [Gaggl and Woltran, 2012]

Given an AF $F = (A, R)$.

$$cf2(F) = \{ S \mid S \in naive(F) \cap naive([F - \Delta_{F,S}]) \}.$$  

Example

$S = \{a, d, e, g, i\}, \ S \in naive(F)$. 

\begin{center}
\begin{tikzpicture}[->,>=stealth,auto]
\node (a) at (0,0) {$a$};
\node (b) at (1,-1) {$b$};
\node (c) at (0,-2) {$c$};
\node (d) at (2,0) {$d$};
\node (e) at (1,-3) {$e$};
\node (f) at (2,-2) {$f$};
\node (g) at (4,0) {$g$};
\node (h) at (4,-2) {$h$};
\node (i) at (2,-3) {$i$};

\path (a) edge (b)
      (a) edge (c)
      (b) edge (c)
      (b) edge (d)
      (b) edge (f)
      (c) edge (e)
      (c) edge (f)
      (d) edge (e)
      (d) edge (g)
      (e) edge (f)
      (f) edge (g)
      (f) edge (i)
      (g) edge (h)
      (h) edge (i);
\end{tikzpicture}
\end{center}
**cf2 Extensions [Gaggl and Woltran, 2012]**

Given an AF $F = (A, R)$.

$$
\text{cf2}(F) = \{ S \mid S \in \text{naive}(F) \cap \text{naive}([[F - \Delta_{F,S}]]) \}. 
$$

**Example**

$S = \{a, d, e, g, i\}, \ S \in \text{naive}(F), \ \Delta_{F,S}(\emptyset) = \{f\}.$

[Diagram of a graph with nodes a, b, c, d, e, f, g, h, i, showing connections between them.]

---

Sarah A. Gaggl, TU Vienna

Comprehensive Analysis of cf2 Semantics
**Alt. Characterization of cf2 ctd.**

**cf2 Extensions [Gaggl and Woltran, 2012]**

Given an AF $F = (A, R)$.

$$cf2(F) = \{ S \mid S \in naive(F) \cap naive([F - \Delta_{F,S}]) \}.$$  

**Example**

$S = \{a, d, e, g, i\}$, $S \in naive(F)$, $\Delta_{F,S}(\{f\}) = \{f, h\}$. 

![Diagram](image-url)
**Alt. Characterization of \( cf_2 \) ctd.**

**cf2 Extensions [Gaggl and Woltran, 2012]**

Given an AF \( F = (A, R) \).

\[
\text{cf2}(F) = \{ S \mid S \in \text{naive}(F) \cap \text{naive}([F - \Delta_{F,S}]) \}.
\]

**Example**

\( S = \{a, d, e, g, i\} \), \( S \in \text{naive}(F) \), \( \Delta_{F,S}(\{f, h\}) = \{f, h\} \).
Alt. Characterization of $cf_2$ ctd.

$cf_2$ Extensions [Gaggl and Woltran, 2012]

Given an AF $F = (A, R)$.

$$cf_2(F) = \{ S \mid S \in naive(F) \cap naive([[F - \Delta_{F,S}]])) \}.$$ 

Example

$S = \{a, d, e, g, i\}$, $S \in naive(F)$, $\Delta_{F,S} = \{f, h\}$, $S \in naive([[F - \Delta_{F,S}]])$, thus $S \in cf_2(F)$. 

![Diagram](image-url)
Alt. Characterization of $\text{cf}_2$ ctd.

$\text{cf}_2$ Extensions [Gaggl and Woltran, 2012]

Given an AF $F = (A, R)$.

$$\text{cf}_2(F) = \{ S \mid S \in \text{naive}(F) \cap \text{naive}([[F - \Delta_{F,S}]]) \}.$$ 

Example

$$\text{cf}_2(F) = \{\{a, d, e, g, i\}, \{c, d, e, g, i\}, \{b, f, h\}, \{b, g, i\}\}.$$
Shortcomings of \( cf^2 \)

- \( cf^2 \) produces questionable results on AFs with cycles of length \( \geq 6 \).

Example

\[
\begin{align*}
\text{cyclic graph:} & \quad \text{cycles: } a \rightarrow b \rightarrow c \rightarrow d \rightarrow e \rightarrow f \rightarrow a \\
\text{Example: } F = \{a, d\}, \{b, e\}, \{c, f\}, \{a, c, e\}, \{b, d, f\}.
\end{align*}
\]

\[
\begin{align*}
\text{cyclic stage: } & \quad \text{cycles: } a \rightarrow c \rightarrow e \rightarrow b \\
\text{cyclic stage: } & \quad \text{cycles: } b \rightarrow d \rightarrow f \rightarrow a \\
\text{Example: } & \quad \text{cycles: } a \rightarrow b \rightarrow c \rightarrow d \rightarrow e \rightarrow f \rightarrow a.
\end{align*}
\]
Shortcomings of \textit{cf}2 and Stage

- \textit{cf}2 produces questionable results on AFs with cycles of length $\geq 6$.
- The grounded extension is not necessarily contained in every stage extension.
  - Stage semantics does not satisfy directionality.

Example

$$\text{stage}(F) = \{\{a\}, \{b\}\} \text{ but } \text{cf}2(F) = \text{ground}(F) = \{\{a\}\}.$$
We combine $\textit{cf2}$ and stage semantics [Dvorák and Gaggl, 2012a], by

1. using the SCC-recursive schema of the $\textit{cf2}$ semantics and

2. instantiate the base case with stage semantics.
We combine \textit{cf2} and stage semantics [Dvorák and Gaggl, 2012a], by

- using the SCC-recursive schema of the \textit{cf2} semantics and
- instantiate the base case with stage semantics.

\textit{stage2} Extensions

For any AF \( F \),

\[
\text{stage2}(F) = \{ S \mid S \in \text{naive}(F) \cap \text{stage}([[F - \Delta_{F,S}]]) \}.
\]
For any AF $F$, $\text{stage2}(F) = \{ S \mid S \in \text{naive}(F) \cap \text{stage}([[F - \Delta F,S]]))\}$. 

**Example**

$$\text{stage2}(F) = \text{cf2}(F) = \{\{a\}\}, \text{ where } \text{stage}(F) = \{\{a\}, \{b\}\}.$$
For any AF \( F \), \( \text{stage2}(F) = \{ S \mid S \in \text{naive}(F) \cap \text{stage}([F - \Delta_{F,S}]) \} \).

Example

\[
\text{stage2}(F) = \text{cf2}(F) = \{ \{a\} \}, \text{ where } \text{stage}(F) = \{ \{a\}, \{b\} \}.
\]

\[
\text{stage2}(G) = \text{stage}(G) = \{ \{a, c, e\}, \{b, d, f\} \}, \text{ but } \text{cf2}(G) = \text{naive}(F) = \{ \{a, d\}, \{b, e\}, \{c, f\}, \{a, c, e\}, \{b, d, f\} \}.
\]
Relations between Semantics

- stable
- stage2
  - stage
  - cf2
- naive
- conflict-free
Argumentation is a dynamic reasoning process.

Which effect has additional information w.r.t. a semantics?

Which information does not contribute to results no matter which changes are performed?

Identification of kernels to remove redundant attacks [Oikarinen and Woltran, 2011].

Definition
Two AFs $F$ and $G$ are strongly equivalent to each other w.r.t. a semantics $\sigma$, in symbols $F \equiv^\sigma_s G$, iff for each AF $H$, $\sigma(F \cup H) = \sigma(G \cup H)$. 
Theorem

For any AFs $F$ and $G$, $F \equiv_{s}^{cf2} G$ iff $F = G$. 
Theorem

For any AFs $F$ and $G$, $F \equiv_{s}^{cf2} G$ iff $F = G$. 

\begin{tikzpicture}
  \node (a) at (0,0) {a};
  \node (cn-1) at (-1,-2) {cn-1};
  \node (cn-2) at (-1,-3) {cn-2};
  \node (c3) at (-1,-4) {c3};
  \node (c2) at (-2,-2) {c2};
  \node (b) at (2,0) {b};
  \node (cn-1') at (3,-2) {cn-1};
  \node (cn-2') at (3,-3) {cn-2};
  \node (c3') at (3,-4) {c3};
  \node (c2') at (2,-2) {c2};

  \draw[->] (a) -- (cn-1);
  \draw[->] (cn-1) -- (cn-2);
  \draw[->] (cn-2) -- (c2);
  \draw[->] (c2) -- (c3);
  \draw[->] (c3) -- (b);
  \draw[->] (a) -- (b);

  \draw[->] (a) -- (cn-1');
  \draw[->] (cn-1') -- (cn-2');</tikzpicture}
Strong Equivalence w.r.t. \( cf2 \)

**Theorem**

*For any AFs \( F \) and \( G \), \( F \equiv_{s}^{cf2} G \) iff \( F = G \).*

\[
H = (A \cup \{d, x, y, z\}, \\
\{(a, a), (b, b), (b, x), (x, a), (a, y), (y, z), (z, a), \\
(d, c) \mid c \in A \setminus \{a, b\}\}).
\]
Theorem

For any AFs $F$ and $G$, $F \equiv_{s}^{cf2} G$ iff $F = G$.

Let $E = \{d, x, z\}$, $E \in cf2(F \cup H)$ but $E \not\in cf2(G \cup H)$. 
Strong Equivalence w.r.t. $cf^2$

**Theorem**

For any AFs $F$ and $G$, $F \equiv_{s}^{cf^2} G$ iff $F = G$.

Let $E = \{d, x, z\}$, $E \in cf^2(F \cup H)$ but $E \notin cf^2(G \cup H)$. 
Strong Equivalence w.r.t. $\text{cf}^2$

**Theorem**

For any AFs $F$ and $G$, $F \equiv_{\text{cf}^2} G$ iff $F = G$.

Let $E = \{d, x, z\}$, $E \in \text{cf}^2(F \cup H)$ but $E \not\in \text{cf}^2(G \cup H)$. 
Theorem

For any AFs $F$ and $G$, $F \equiv_{s}^{cf2} G$ iff $F = G$.

Let $E = \{d, x, z\}$, $E \in cf2(F \cup H)$ but $E \notin cf2(G \cup H)$. 
Theorem

For any AFs $F$ and $G$, $F \equiv^c_f G$ iff $F = G$.

Let $E = \{d, x, z\}$, $E \in cf2(F \cup H)$ but $E \notin cf2(G \cup H)$. 

Sarah A. Gaggl, TU Vienna

Comprehensive Analysis of cf2 Semantics

FACULTY OF INFORMATICS
Theorem

For any AFs $F$ and $G$, $F \equiv^{cf2}_s G$ iff $F = G$.

Let $E = \{d, x, z\}$, $E \in cf2(F \cup H)$ but $E \not\in cf2(G \cup H)$. 
**Theorem**

For any AFs $F$ and $G$, $F \equiv_{s}^{cf2} G$ iff $F = G$.

Let $E = \{d, x, z\}$, $E \in cf2(F \cup H)$ but $E \notin cf2(G \cup H)$. 
Theorem

For any AFs $F$ and $G$, $F \equiv_{s}^{cf2} G$ iff $F = G$.

No matter which AFs $F \neq G$, one can always construct an $H$ s.t.
$cf2(F \cup H) \neq cf2(G \cup H)$;
Theorem

For any AFs $F$ and $G$, $F \equiv_{s}^{cf2} G$ iff $F = G$.

- No matter which AFs $F \neq G$, one can always construct an $H$ s.t. $cf2(F \cup H) \neq cf2(G \cup H)$;
- For stage2 semantics also strong equivalence coincides with syntactic equivalence.
No matter which AFs $F \neq G$, one can always construct an $H$ s.t.
$\text{cf2}(F \cup H) \neq \text{cf2}(G \cup H)$;

For stage2 semantics also strong equivalence coincides with syntactic equivalence.

**Succinctness Property**

An argumentation semantics $\sigma$ satisfies the **succinctness property** or is **maximal succinct** iff no AF contains a redundant attack w.r.t. $\sigma$. 
### Complexity Analysis

<table>
<thead>
<tr>
<th></th>
<th>Ver</th>
<th>Cred</th>
<th>Skept</th>
<th>Exists(^{-\emptyset})</th>
</tr>
</thead>
<tbody>
<tr>
<td>naive</td>
<td>in P</td>
<td>in P</td>
<td>in P</td>
<td>in P</td>
</tr>
<tr>
<td>stable</td>
<td>in P</td>
<td>NP-c</td>
<td>coNP-c</td>
<td>NP-c</td>
</tr>
<tr>
<td>cf2</td>
<td>in P</td>
<td>NP-c</td>
<td>coNP-c</td>
<td>in P</td>
</tr>
<tr>
<td>stage</td>
<td>coNP-c</td>
<td>(\Sigma^P_2)-c</td>
<td>(\Pi^P_2)-c</td>
<td>in P</td>
</tr>
<tr>
<td>stage2</td>
<td>coNP-c</td>
<td>(\Sigma^P_2)-c</td>
<td>(\Pi^P_2)-c</td>
<td>in P</td>
</tr>
</tbody>
</table>

**Table:** Computational complexity of naive-based semantics.
Implementation

Reduction-based Approach

- Answer-set Programming (ASP) encodings for \textit{cf2} and \textit{stage2}.
- Saturation vs. \texttt{metasp} encodings for \textit{stage2}.
- All encodings incorporated in the system \textsc{ASPARTIX} [Egly et al., 2010].

Direct Approach

- Labeling-based algorithms for \textit{cf2} and \textit{stage2}.

Web-Application

http://rull.dbai.tuwien.ac.at:8080/ASPARTIX
Conclusion

- **Alternative characterization** for $cf_2$ to avoid the recursive computation of sub-frameworks.
- **stage2** semantics overcomes problems of $cf_2$.
- **Strong equivalence** w.r.t. $cf_2$ (resp. stage2) coincides with syntactic equivalence.
- Provided the missing complexity results for $cf_2$ (resp. stage2).
- Implementation in terms of ASP and labeling-based algorithms.
Future Work

- Further relations to other semantics like intertranslatability.
- Optimizations of ASP encodings.
- Development of appropriate instantiation methods for naive-based semantics.
- Other combinations of semantics in the alternative characterization, like \( \text{sem}(F) = \{ S \mid \sigma(F) \cap \tau([[F - \Delta_{F,S}]])) \}. \)
On principle-based evaluation of extension-based argumentation semantics.

Scc-recursiveness: A general schema for argumentation semantics.

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

Incorporating stage semantics in the scc-recursive schema for argumentation semantics.

Computational aspects of cf2 and stage2 argumentation semantics.

Answer-set programming encodings for argumentation frameworks.

The cf2 argumentation semantics revisited.

Oikarinen, E. and Woltran, S. (2011)
Characterizing strong equivalence for argumentation frameworks.
<table>
<thead>
<tr>
<th></th>
<th>$\text{cf2}$</th>
<th>$\text{stage2}$</th>
<th>$\text{stable}$</th>
<th>$\text{stage}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Cred}_\sigma^{\text{acycl}}$</td>
<td>in P</td>
<td>in P</td>
<td>P-c</td>
<td>P-c</td>
</tr>
<tr>
<td>$\text{Skept}_\sigma^{\text{acycl}}$</td>
<td>in P</td>
<td>in P</td>
<td>P-c</td>
<td>P-c</td>
</tr>
<tr>
<td>$\text{Cred}_\sigma^{\text{even-free}}$</td>
<td>NP-c</td>
<td>coNP-h</td>
<td>P-c</td>
<td>$\Sigma_2^P$-c</td>
</tr>
<tr>
<td>$\text{Skept}_\sigma^{\text{even-free}}$</td>
<td>coNP-c</td>
<td>coNP-h</td>
<td>P-c</td>
<td>$\Pi_2^P$-c</td>
</tr>
<tr>
<td>$\text{Cred}_\sigma^{\text{bipart}}$</td>
<td>in P</td>
<td>in P</td>
<td>P-c</td>
<td>P-c</td>
</tr>
<tr>
<td>$\text{Skept}_\sigma^{\text{bipart}}$</td>
<td>in P</td>
<td>in P</td>
<td>P-c</td>
<td>P-c</td>
</tr>
<tr>
<td>$\text{Cred}_\sigma^{\text{sym}}$</td>
<td>in P</td>
<td>in P/$\Sigma_2^P$*</td>
<td>in P</td>
<td>in P/$\Sigma_2^P$*</td>
</tr>
<tr>
<td>$\text{Skept}_\sigma^{\text{sym}}$</td>
<td>in P</td>
<td>in P/$\Pi_2^P$*</td>
<td>in P</td>
<td>in P/$\Pi_2^P$*</td>
</tr>
</tbody>
</table>

**Table:** Complexity results for special AFs (* with self-attacking arguments).