The Second International Competition on Computational Models of Argumentation (ICCMA'17)*

Solver Requirements

Sarah A. Gaggl Thomas Linsbichler Marco Maratea Stefan Woltran

October 19, 2016

This document contains requirements for solvers participating at ICCMA'17. In particular, this document provides some formal background on abstract argumentation, a list of computation problems (tasks) considered in the competition, the input formats of benchmark instances, the expected output format of results, and a description of the interface participating solvers are required to provide.

1 Abstract Argumentation

An *abstract argumentation framework* (AF, for short) (Dung, 1995) is a tuple $\mathcal{F} = (\mathsf{A}, \rightarrow)$ where A is a set of arguments and $\rightarrow$ is a relation $\rightarrow \subseteq \mathsf{A} \times \mathsf{A}$. For two arguments $a, b \in \mathsf{A}$ the relation $a \rightarrow b$ means that argument a *attacks* argument b. An argument $a \in \mathsf{A}$ is *defended* by $S \subseteq \mathsf{A}$ (in $\mathcal{F}$) if for each $b \in \mathsf{A}$ such that $b \rightarrow a$ there is some $c \in S$ such that $c \rightarrow b$. A set $E \subseteq \mathsf{A}$ is *conflict-free* (in $\mathcal{F}$) if and only if there are no $a, b \in E$ with $a \rightarrow b$. E is *admissible* (in $\mathcal{F}$) if and only if it is conflict-free and each $a \in E$ is defended by E. Finally, the range of E (in $\mathcal{F}$) is given by $E^+_\mathcal{F} = E \cup \{a \in \mathsf{A} \mid \exists b \in E : b \rightarrow a\}$.

Semantics are used to determine sets of jointly acceptable arguments by mapping each AF $\mathcal{F} = (\mathsf{A}, \rightarrow)$ to a set of extensions $\sigma(\mathcal{F}) \subseteq 2^\mathsf{A}$. The extensions under complete, preferred, stable, semi-stable (Caminada *et al.*, 2012), stage (Verheij, 1996), grounded and ideal (Dung *et al.*, 2007) semantics are defined as follows. Given an AF $\mathcal{F} = (\mathsf{A}, \rightarrow)$ and a set $E \subseteq \mathsf{A}$,

- $E \in \mathbf{CO}(\mathcal{F})$ iff E is admissible in $\mathcal{F}$ and if $a \in \mathsf{A}$ is defended by E in $\mathcal{F}$ then $a \in E$,
- $E \in \mathbf{PR}(\mathcal{F})$ iff $E \in \mathbf{CO}(\mathcal{F})$ and there is no $E' \in \mathbf{CO}(\mathcal{F})$ s.t. $E' \supset E$,
- $E \in \mathbf{ST}(\mathcal{F})$ iff $E \in \mathbf{CO}(\mathcal{F})$ and $E^+_\mathcal{F} = \mathsf{A}$,
- $E \in \mathbf{SST}(\mathcal{F})$ iff $E \in \mathbf{CO}(\mathcal{F})$ and there is no $E' \in \mathbf{CO}(\mathcal{F})$ s.t. $E'^+_\mathcal{F} \supset E^+_\mathcal{F}$,
- $E \in \mathbf{STG}(\mathcal{F})$ iff E is conflict-free in $\mathcal{F}$ and there is no E' that is conflict-free in $\mathcal{F}$ s.t. $E'^+_\mathcal{F} \supset E^+_\mathcal{F}$,
- $E \in \mathbf{GR}(\mathcal{F})$ iff $E \in \mathbf{CO}(\mathcal{F})$ and there is no $E' \in \mathbf{CO}(\mathcal{F})$ s.t. $E' \subset E$,

*Adapted from last edition's version by Federico Cerutti, Nir Oren, Hannes Straß, Matthias Thimm, and Mauro Vallati

- $E \in \mathbf{ID}(\mathcal{F})$ if and only if E is admissible, $E \subseteq \bigcap \mathbf{PR}(\mathcal{F})$ and there is no admissible $E' \subseteq \bigcap \mathbf{PR}(\mathcal{F})$ s.t. $E' \supset E$.

For more discussion on these semantics see (Baroni *et al.*, 2011).

Note that both grounded and ideal extensions are uniquely determined and always exist (Dung, 1995; Dung *et al.*, 2007). Thus, they are also called *single-status* semantics. The other semantics introduced are *multi-status* semantics. That is, there is not always a unique extension induced by the semantics. In order to reason with multi-status semantics, usually, one takes either a credulous or skeptical perspective.

That is, given an AF $\mathcal{F} = (\mathsf{A}, \rightarrow)$ and a semantics $\sigma \in \{\mathbf{CO}, \mathbf{PR}, \mathbf{ST}, \mathbf{SST}, \mathbf{STG}, \mathbf{GR}, \mathbf{ID}\}$, argument $a \in \mathsf{A}$ is

- *credulously accepted* in $\mathcal{F}$ under semantics σ if there is a σ-extension $E \in \sigma(\mathcal{F})$ with $a \in E$, and
- *skeptically accepted* in $\mathcal{F}$ with semantics σ if for all σ-extensions $E \in \sigma(\mathcal{F})$ it holds that $a \in E$.

Recall that stable semantics is the only case where an AF might possess no extension. In such a situation, each argument is defined to be skeptically accepted.

2 Computational problems

Let $\sigma \in \{\mathbf{CO}, \mathbf{PR}, \mathbf{ST}, \mathbf{SST}, \mathbf{STG}, \mathbf{GR}, \mathbf{ID}\}$ be some semantics. We consider the following computational tasks[1]:

DC-σ **Given** $\mathcal{F} = (\mathsf{A}, \rightarrow)$ and $a \in \mathsf{A}$, **decide** whether a is credulously accepted in $\mathcal{F}$ under σ.

DS-σ **Given** $\mathcal{F} = (\mathsf{A}, \rightarrow)$ and $a \in \mathsf{A}$, **decide** whether a is skeptically accepted in $\mathcal{F}$ under σ.

SE-σ **Given** $\mathcal{F} = (\mathsf{A}, \rightarrow)$, **return** some set $E \subseteq \mathsf{A}$ that is a σ-extension of $\mathcal{F}$.

EE-σ **Given** $\mathcal{F} = (\mathsf{A}, \rightarrow)$, **enumerate** all sets $E \subseteq \mathsf{A}$ that are σ-extensions of $\mathcal{F}$.

Moreover, a special track called *Dung's Triathlon* is conducted:

D3 **Given** $\mathcal{F} = (\mathsf{A}, \rightarrow)$, **enumerate**

- all sets $E \subseteq \mathsf{A}$ that are **GR**-extensions of $\mathcal{F}$, followed by
- all sets $E \subseteq \mathsf{A}$ that are **ST**-extensions of $\mathcal{F}$, followed by
- all sets $E \subseteq \mathsf{A}$ that are **PR**-extensions of $\mathcal{F}$.

3 Input File formats

Each benchmark instance will be provided in two different file formats: trivial graph format (`tgf`) and aspartix format (`apx`). In the following we present the AF $\mathcal{F} = (\mathsf{A}, \rightarrow)$ where $\mathsf{A} = \{a1, a2, a3\}$ and $\rightarrow = \{(a1, a2), (a2, a3), (a2, a1)\}$ in each of these formats.

[1] With the exception of **DS-GR**, **EE-GR**, **DS-ID**, and **EE-ID**.

3.1 Trivial Graph Format

See `http://en.wikipedia.org/wiki/Trivial_Graph_Format`. In the following we will refer to the file containing this instance by `myFile.tgf`.

```
1
2
3
#
1 2
2 3
2 1
```

3.2 Aspartix Format

See (Egly *et al.*, 2010). In the following we will refer to the file containing this instance by `myFile.apx`.

```
arg(a1).
arg(a2).
arg(a3).
att(a1,a2).
att(a2,a3).
att(a2,a1).
```

4 Output Format

Solvers must write the result to standard output exactly in the format described below.

- **DC**-σ for $\sigma \in \{\mathbf{CO}, \mathbf{PR}, \mathbf{ST}, \mathbf{SST}, \mathbf{STG}, \mathbf{GR}, \mathbf{ID}\}$:
  The output must be either

  ```
  YES
  ```

  expressing that the queried argument is credulously accepted in the given AF under σ, or

  ```
  NO
  ```

  if the queried argument is not credulously accepted in the given AF under σ.
- **DS**-σ for $\sigma \in \{\mathbf{CO}, \mathbf{PR}, \mathbf{ST}, \mathbf{SST}, \mathbf{STG}\}$:
  The output must be either

  ```
  YES
  ```

  expressing that the queried argument is skeptically accepted in the given AF under σ, or

  ```
  NO
  ```

if the queried argument is not skeptically accepted in the given AF under σ.

- **SE**-σ for $\sigma \in \{\mathbf{CO}, \mathbf{PR}, \mathbf{ST}, \mathbf{SST}, \mathbf{STG}, \mathbf{GR}, \mathbf{ID}\}$:
  The output must be of the form

  ```
  [a1,a3,a6]
  ```

  expressing that $\{a1, a3, a6\}$ is a σ-extension of the given AF, or

  ```
  NO
  ```

  expressing that there does not exist a σ-extension of the given AF.

- **EE**-σ for $\sigma \in \{\mathbf{CO}, \mathbf{PR}, \mathbf{ST}, \mathbf{SST}, \mathbf{STG}\}$:
  The output must be of the form

  ```
  [[a1,a2],[a1,a3],[a2,a3]]
  ```

  expressing that the set of σ-extensions of the given AF is exactly $\{\{a1, a2\}, \{a1, a3\}, \{a2, a3\}\}$, or

  ```
  []
  ```

  expressing that there does not exist a σ-extension of the given AF. Note that if the empty set is the only σ-extension of the given AF, the output is required to be `[[]]`.

- **D3**:
  The output must be of the form

  ```
  [[a1]],[],[[a1,a2],[a1,a3]]
  ```

  expressing that the grounded extension of the given AF is $\{a1\}$, the given AF has no stable extensions, and that its set of preferred extensions is exactly $\{\{a1, a2\}, \{a1, a3\}\}$.

5 Solver Interface

The single executable of a solver should be runnable from a command line and must provide the following behavior (let `solver` be the filename of the executable):

- `solver` (without any parameters)
  Prints author and version information of the solver on standard output.
  Example:

  ```
  user$ solver
  MySolver v1.0
  John Smith
  user$ _
  ```

- `solver --formats`
  Prints the supported formats of the solver in the form

  `[supportedFormat1,supportedFormat2, ..., supportedFormatN]`

  The possible values for each supported format are `tgf`, `apx`.
  Example:

  ```
  user$ solver --formats
  [tgf,apx]
  user$ _
  ```

- `solver --problems`
  Prints the supported computational problems in the form

  `[supportedProblem1,supportedProblem2, ..., supportedProblemN]`

  The possible values for each supported problem are `DC-CO`, `DC-PR`, `DC-ST`, `DC-SST`, `DC-STG`, `DC-GR`, `DC-ID`, `DS-CO`, `DS-PR`, `DS-ST`, `DS-SST`, `DS-STG`, `SE-CO`, `SE-PR`, `SE-ST`, `SE-SST`, `SE-STG`, `SE-GR`, `SE-ID`, `EE-CO`, `EE-PR`, `EE-ST`, `EE-SST`, `EE-STG`, and `D3`.
  Example:

  ```
  user$ solver --problems
  [DC-CO,DS-CO,EE-CO,SE-ST]
  user$ _
  ```

- `solver -p <task> -f <file> -fo <fileformat> [-a <additional_parameter>]`
  Solves the given problem on the argumentation framework specified by the given file (represented in the given file format) and prints out the result. More specifically:
  - `solver -p DC-<semantics> -f <file> -fo <fileformat> -a <additional_parameter>`
    Solves the problem of deciding whether an argument (given as additional parameter) is credulously inferred and prints out either `YES` (if it is credulously inferred) or `NO` (if it is not credulously inferred).
    Example:

    ```
    user$ solver -p DC-CO -f myFile.apx -fo apx -a a1
    YES
    user$ _
    ```

  - `solver -p DS-<semantics> -f <file> -fo <fileformat> -a <additional_parameter>`
    Solves the problem of deciding whether an argument (given as additional parameter) is skeptically inferred and prints out either `YES` (if it is credulously inferred) or `NO` (if it is not credulously inferred).
    Example:

    ```
    user$ solver -p DC-ST -f myFile.apx -fo apx -a a2
    NO
    user$ _
    ```

- `solver -p SE-<semantics> -f <file> -fo <fileformat>`
  Returns one extension wrt. the given semantics in the format `[A1,A2,...,AN]` (no further parameters needed).
  Example:

  ```
  user$ solver -p SE-PR -f myFile.tgf -fo tgf
  [a1,a3]
  user$ _
  ```

  If there does not exist any extension wrt. the given semantics (which can be the case for stable semantics) then the output `NO` is expected by the solver.
  Example:

  ```
  user$ solver -p SE-ST -f anotherFile.tgf -fo tgf
  NO
  user$ _
  ```

- `solver -p EE-<semantics> -f <file> -fo <fileformat>`
  Enumerates all sets that are extensions wrt. the given semantics in the format `[[A1,A2,...,AN],[B1,B2,...,BM],..., [Z1,Z2,...,ZN]]` (no further parameters needed).
  Example:

  ```
  user$ solver -p EE-PR -f myFile.apx -fo apx
  [[a1,a3],[a2]]
  user$ _
  ```

- `solver -p D3 -f <file> -fo <fileformat>`
  Enumerates all sets that are extensions wrt. **GR**, followed by all sets that are extensions wrt. **ST**, followed by all sets that are extensions w.r.t. **PR**, each in the same format as above. (no further parameters needed).
  Example:

  ```
  user$ solver -p D3 -f myFile.apx -fo apx
  [[]],[[a1,a3],[a2]],[[a1,a3],[a2]]
  user$ _
  ```

Each solver has to support at least one file format. It is ensured that each solver is only called with file formats and problems he supports (the behavior of a solver when called with unsupported parameters is undefined).

References

P. Baroni, M. Caminada, and M. Giacomin. An introduction to argumentation semantics. *The Knowledge Engineering Review*, 26(4):365–410, 2011.

M. W. A. Caminada, W. A. Carnielli, and P. E. Dunne. Semi-stable semantics. *J. Log. Comput.*, 22(5):1207–1254, 2012.

P. M. Dung, P. Mancarella, and F. Toni. Computing ideal sceptical argumentation. *Artificial Intelligence*, 171(10–15):642–674, 2007.

P. M. Dung. On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games. *Artificial Intelligence*, 77(2):321–358, 1995.

U. Egly, S. A. Gaggl, and S. Woltran. Answer-set programming encodings for argumentation frameworks. *Argument & Computation*, 1(2):147–177, 2010.

B. Verheij. Two approaches to dialectical argumentation: admissible sets and argumentation stages. In *Proceedings of the 8th Dutch Conference on Artificial Intelligence (NAIC'96)*, pages 357–368, 1996.