Towards a framework for the implementation and verification of translations between argumentation models

Bas van Gijzel
University of Nottingham

### Argumentation theory

Interdisciplinary area with various applications:
- **Law**: Modelling legal problems/cases
- **Decision making**: Organising (inconsistent) information
- **Communication theory**: Making texts precise

- Different notions of argument
- Different argumentation models

### Abstract argumentation as a translation target

Dung’s argumentation frameworks are a gold standard:
- Most other models are an instantiation of Dung’s model
- Relatively simple data structures/algorithms
- Recent efforts to optimise the evaluation of AF’s

### A perceived problem

- Lack of implementations of complex models
- Existing translations to Dung, however again a lack of implementations:
  1. Translations are complex
  2. Proofs of correctness are complex (page long proofs)

### Implementing Dung’s model

An abstract argumentation framework (AF) is a tuple \( AF = \langle \text{Args}, \text{Def} \rangle \) such that:
- \( \text{Args} \) is a set of (abstract) arguments;
- \( \text{Def} \subseteq \text{Args} \times \text{Args} \)

\[ A \rightarrow B \rightarrow C \]

**Conflict-freeness:**
A set \( S \subseteq \text{Args} \) of arguments is called conflict-free iff there is no \( A, B \in S \) such that \( (A, B) \in \text{Def} \)

\[ \text{DungAF arg} = \text{AF} [\text{arg}] [(\text{arg}, \text{arg})] \]

**Type**
\[ \text{AbsArg} = \text{String} \]

\[ a, b, c :: \text{AbsArg} \]
\[ a = "A" \]
\[ b = "B" \]
\[ c = "C" \]

\[ \text{AF}_1 :: \text{DungAF AbsArg} \]
\[ \text{AF}_1 = \text{AF} [a, b, c] [(a, b), (b, c)] \]

\[ \text{conflictFree :: Eq arg} \Rightarrow \text{DungAF arg} \rightarrow \text{[arg]} \rightarrow \text{Bool} \]
\[ \text{conflictFree (AF – def) args} = \text{null} [(a, b) | (a, b) ↔ def, a ∈ args, b ∈ args] \]

### Overview of work done

**Haskell**:
- Implementation of Dung’s AFs
- Implementation of Carneades
- All code documented, available as:
  - Literate Haskell
  - Cabal package
- Sketch of translation and its derived properties for: Carneades → Dung

**Agda**:
- Formalisation of Dung’s AFs
- All code documented, available as literate Agda

### Proposed solution

**Haskell implementation of models**:
- **High-level code** close to the mathematical definitions of argumentation models, implemented in a tutorial-like fashion, giving:
  1. Increased insight into the implementation of argumentation models
  2. Easier implementation of existing/future translations

**Agda formalisation of models**:
- **High-level code**, formally implementing argumentation models giving:
  1. Automatically verified code
  2. Easier formalisation of existing/future translations

**Implementation and formalisation of translations**:
- Implementation in Haskell and formalisation of that translation in Agda giving:
  1. Access to existing efficient tools for the translation target
  2. A mechanically proven/sound way to translate from model to model

### Future work

- Further formalisation of Dung’s definitions and theorems:
  - Formalisation of fixpoints and general lemmas
- Implementation and formalisation of the translation from Carneades to Dung.
- Connect the implementation of Dung’s AFs to an optimised implementation using ASP or SAT