The Pyglaf Argumentation Reasoner

Mario Alviano

Department of Mathematics and Computer Science, University of Calabria, Italy
alviano@mat.unical.it

Abstract

The pyglaf reasoner takes advantage of circumscription to solve computational problems of abstract argumentation frameworks. In fact, many of these problems are reduced to circumscription by means of linear encodings, and a few others are solved by means of a sequence of calls to an oracle for circumscription. Within pyglaf, Python is used to build the encodings and to control the execution of the external circumscription solver, which extends the SAT solver GLUCOSE and implements an algorithm based on unsatisfiable core analysis.

1998 ACM Subject Classification
I.2.4 Knowledge Representation Formalisms and Methods

Keywords and phrases abstract argumentation frameworks, propositional circumscription, minimal model enumeration, incremental solving

Digital Object Identifier 10.4230/OASIcs.ICLP.2017.2

1 Introduction

Circumscription [9] is a nonmonotonic logic formalizing common sense reasoning by means of a second order semantics, which essentially enforces to minimize the extension of some predicates. With a little abuse on the definition of circumscription, the minimization can be imposed on a set of literals, so that a set of negative literals can be used to encode a maximization objective function. Since many semantics of abstract argumentation frameworks are based on a preference relation that essentially amount to inclusion relationships, pyglaf (http://alviano.com/software/pyglaf/) uses circumscription as a target language to solve computational problems of abstract argumentation frameworks.

PYGLAF is implemented in Python and uses CIRCUMSCRIPTINO [1], a circumscription solver extending the SAT solver GLUCOSE [7] with the unsatisfiable core based algorithm one [6] enhanced by reiterated progression shrinking [3], native support for cardinality constraints as in wasp [4, 5, 8], and polyspace model enumeration [2]. Linear reductions are used for all considered semantics. The communication between pyglaf and CIRCUMSCRIPTINO is handled in the simplest possible way, that is, via stream processing. In fact, the communication is limited to a single invocation of the circumscription solver.

2 From Argumentation Frameworks to Circumscription

Let \( A \) be a fixed, countable set of atoms including \( \bot \). A literal \( \ell \) is an atom possibly preceded by the connective \( \neg \). For a literal \( \ell \), let \( \overline{\ell} \) denote its complementary literal, that is, \( p = \neg p \) and \( \overline{\overline{p}} = p \) for all \( p \in A \); for a set \( L \) of literals, let \( \mathcal{T} \) be \( \{ \overline{\ell} \mid \ell \in L \} \). Formulas are defined as

* The paper has been partially supported by the Italian Ministry for Economic Development (MISE) under project “PIUCultura – Paradigmi Innovativi per l’Utilizzo della Cultura” (n. F/020016/01-02/X27), and under project “Smarter Solutions in the Big Data World (S2BDW)” (n. F/056389/01-03/X32) funded within the call “HORIZON2020” PON I&I C 2014-2020, and by Gruppo Nazionale per il Calcolo Scientifico (GNCS-INdAM).
usual by combining atoms and the connectives \( \neg, \land, \lor, \rightarrow, \leftrightarrow \). A \textit{theory} is a set \( T \) of formulas including \( \neg \bot \); the set of atoms occurring in \( T \) is denoted by \( \text{atoms}(T) \). An \textit{assignment} is a set \( A \) of literals such that \( A \cap \overline{A} = \emptyset \). An \textit{interpretation} for a theory \( T \) is an assignment \( I \) such that \( (I \cup \overline{T}) \cap A = \text{atoms}(T) \). Relation \( \models \) is defined as usual. \( I \) is a \textit{model} of a theory \( T \) if \( I \models T \). Let \( \text{models}(T) \) denote the set of models of \( T \).

\textit{Circumscription} applies to a theory \( T \) and a set \( P \) of literals subject to minimization. Formally, relation \( \leq^P \) is defined as follows: for \( I, J \) interpretations of \( T \), \( I \leq^P J \) if \( I \cap P \subseteq J \cap P \). \( I \in \text{models}(T) \) is a \textit{preferred model} of \( T \) with respect to \( \leq^P \) if there is no \( J \in \text{models}(T) \) such that \( I \not\leq^P J \) and \( J \leq^P I \). Let \( \text{CIRC}(T, P) \) denote the set of preferred models of \( T \) with respect to \( \leq^P \).

An \textit{abstract argumentation framework} (AF) is a directed graph \( G \) whose nodes \( \text{arg}(G) \) are arguments, and whose arcs \( \text{att}(G) \) represent an attack relation. An \textit{extension} \( E \) is a set of arguments. The \textit{range} of \( E \) in \( G \) is \( E^+_G := E \cup \{ x \mid \exists y x \in \text{att}(G) \text{ with } y \in E \} \). In the following, several AF semantics are characterized by means of circumscription.

For each argument \( x \), an atom \( a_x \) is possibly introduced to represent that \( x \) is attacked by some argument that belongs to the computed extension \( E \), and an atom \( r_x \) is possibly introduced to enforce that \( x \) belongs to the range \( E^+_G \):

\[
\text{attacked}(G) := \left\{ a_x \leftrightarrow \bigvee_{yx \in \text{att}(G)} y \mid x \in \text{arg}(G) \right\}
\]

\[
\text{range}(G) := \left\{ r_x \rightarrow x \vee \bigvee_{yx \in \text{att}(G)} y \mid x \in \text{arg}(G) \right\}
\]

The following set of formulas characterize semantics not based on preferences:

\[
\text{conflict-free}(G) := \{ \neg \bot \} \cup \{ \neg x \lor \neg y \mid xy \in \text{att}(G) \}
\]

\[
\text{admissible}(G) := \text{conflict-free}(G) \cup \text{attacked}(G) \cup \{ x \rightarrow a_y \mid yx \in \text{att}(G) \}
\]

\[
\text{complete}(G) := \text{admissible}(G) \cup \left\{ \left( \bigwedge_{yx \in \text{att}(G)} a_y \right) \rightarrow x \mid x \in \text{arg}(G) \right\}
\]

\[
\text{stable}(G) := \text{complete}(G) \cup \text{range}(G) \cup \{ r_x \mid x \in \text{arg}(G) \}
\]

Note that in (4) truth of an argument \( x \) implies that all arguments attacking \( x \) are actually attacked by some true argument. In (5), instead, whenever all attackers of an argument \( x \) are attacked by some true argument, argument \( x \) is forced to be true. Finally, in (6) all atoms of the form \( r_x \) are forced to be true, so that the range of the computed extension has to cover all arguments.

Below are several AF semantics with natural characterization in circumscription:

\[
\text{co}(G) := \text{CIRC}(\text{complete}(G), \emptyset)
\]

\[
\text{st}(G) := \text{CIRC}(\text{stable}(G), \emptyset)
\]

\[
\text{gr}(G) := \text{CIRC}(\text{complete}(G), \text{arg}(G))
\]

\[
\text{pr}(G) := \text{CIRC}(\text{complete}(G), \overline{\text{arg}(G)})
\]

\[
\text{sst}(G) := \text{CIRC}(\text{complete}(G) \cup \text{range}(G), \{ \neg r_x \mid x \in \text{arg}(G) \})
\]

\[
\text{stg}(G) := \text{CIRC}(\text{conflict-free}(G) \cup \text{range}(G), \{ \neg r_x \mid x \in \text{arg}(G) \})
\]

All of the above semantics are supported by \textsc{pyglaf}, which provides a uniform developing platform for reasoning on argumentation frameworks.
References


