Arguments and Artifacts for Dispute Resolution

Enrico Oliva  Mirko Viroli  Andrea Omicini

ALMA MATER STUDIORUM–Università di Bologna, Cesena, Italy

WOA 2008 – Palermo, Italy, 18th November 2008
Outline

1 Motivation/Background
   - Motivation
   - Architecture for ADR

2 Argumentation and Persuasion
   - Argumentation System
   - Dialogue System
   - Persuasion Dialogue Protocol
   - Example of Run

3 Conclusions
Argumentation Theory

Actually, argumentation is . . .

- a formal discipline within Artificial Intelligence whose aim is to make a computer assist in or perform the act of argumentation

Argumentation is useful

- where formal logic and classical decision theory are unable to capture the richness of reasoning
  - in complex software systems in order to represent conflicting knowledge
  - in the construction of systems for legal reasoning
  - in Multi-agent Systems to model the **communication** between agents

. . . **good communication** can help to overcome and resolve most of the problems
Argumentation and Dialogue

Dialogue is a reciprocal conversation between two or more people/agents
  could be modelled essentially as a dialectical exchange of arguments

Argumentation supports dialogue in MAS in order to
  exchange information, resolve disputes and persuade each other

[Walton and Krabbe, 1995] define six types of dialogues among two entities: persuasion, inquiry, negotiation, information seeking, deliberation and eristic
Arguments and Artifacts for Dispute Resolution

Motivation
Arguments and Alternative Dispute Resolution

Alternative Dispute Resolution (ADR) is an alternative to litigation (negotiation, mediation, collaborative law, and arbitration)

- Arguments have a central role in the process of formal legal systems and also in the trial
- Disputants use arguments in order to persuade the other parts: e.g., decision makers—juries, judges, clients and attorneys
- Online Dispute Resolution is an online instance of ADR system moving it to virtual environment and providing computation and communication support
Objectives

The goal of this paper is to provide an intelligent mediator service for Alternative Dispute Resolution system

To this end we exploit two conceptual frameworks

1. A central co-ordinating entity for argumentative reasoning operation, called *Co-Argumentation Artifact (CAA)* introduced by [Oliva et al., 2008a]

2. A central dialogue entity for communication and reasoning with argument, called *Dialog Artifact (DA)* introduced by [Oliva et al., 2008b]
The general architecture of our ADR system follows the A&A meta model [Omicini et al., 2008]

- the local $CAA_1$ and $CAA_2$ are used by agents in order to coordinate their mental state
- global DA and CAA provide services and functionalities for the entire agent society
  - to provide services for coordination and communication based on arguments
Co-Argumentation Artifact provides co-ordination services to agents, allowing

- to share, store and exchange arguments as a commitment store
- to automatically calculate argument and belief acceptability according to the agent attitudes (credulous, cautious and skeptical)

Definition (Co-Argumentation Artifact)

Co-Argumentation Artifact (CAA) as an artifact specialized in managing arguments and providing coordination services for argumentation process in a MAS
Co-Argumentation Artifact II

List of operation provided by CAA

- **acceptable(Arg, Attitude):** CAA verifies Arg acceptance in the commitment store with specified Attitude
- **read(ArgTemplate):** CAA returns an argument that logically unifies with ArgTemplate
- **conflict(Arg):** CAA verifies the existence of an argument in CAA in rebuttal relation with Arg
- **attack(Arg):** CAA verifies that Arg is in undercut relation with an argument in CAA
- **defeat(Arg):** CAA verifies the existence of an argument in CAA in undercut relation with Arg
- **commit(Arg):** CAA stores Arg and it recompute conflict free sets, admissible sets and preferred extensions
Dialog Artifact (DA) is the abstraction encapsulating the rules of dialogue during persuasion process
Dialog Artifact II

Definition (Dialogue Artifact)

A *Dialogue Artifact* is a triple $DA = \langle DP, CS, IC \rangle$, where

- $DP$ is a collection of specifications of dialogue protocols
  - agents make utterances according to the permitted sequences defined by the protocol specification
- $CS$ is a collection of commitment stores
  - they could be private and public for each participant, together with a central for the dialogue as a whole
- $IC$ is a collection of specifications of interaction controls (IC)
  - it identifies which constraints on the future course of dialogues are created by the existing commitments
Dialog Artifact III

DA suggests agents the admissible moves constrained by the state of the commitment store.

DA provides the following operations:

- \texttt{nextlocutions([L])}: DA returns the list of currently admissible locutions.
- \texttt{lastlocution(L)}: DA returns the last locutions.
- \texttt{state(S)}: DA returns the protocol state.
- \texttt{act(L)}: DA stores locution \( L \) and updates the protocol state.
- \texttt{cs(A)}: DA executes an action \( A \) over the commitment store.
Alternative Dispute Resolution

The idea

We aim at exploiting our framework CAA & DA as a computation and communication support for conflict resolution in an agent-based society.

- In the DA we store the arbitration, mediation or negotiation protocol. The parties exploit the DA to take part in the discussion, which drives the dialogue grounded on the commitments.
  - The advantages are: the management of dialogue between multiple entities and the automatic interaction with commitment/argument store.
- The CAA provides the most suitable abstraction for a commitment/argument store where to evaluate automatically argument validity with respect to a normative context.
Outline

1. Motivation/Background
   - Motivation
   - Architecture for ADR

2. Argumentation and Persuasion
   - Argumentation System
   - Dialogue System
   - Persuasion Dialogue Protocol
   - Example of Run

3. Conclusions
Our Argumentation Approach

The idea

Arguments are built using a monotonic logic (with first order language) and non-monotonicity is expressed at the meta level in terms of interaction between conflicting arguments.

- Abstract Argumentation framework [Dung, 1995] – meta level
- Internal structure of argument express with FOL language with tuple notation
Argument Representation

**Argument** is a sequence of inferences that leads to a conclusion. It is composed of

- **beliefs** are facts and rules that represent premises
- **inference rules** are labels that represent inference processes such as deduction or induction
- **conclusions** are facts that represent results of the inference process applied to the beliefs

**Socrates Argument example**

```
all men are mortal, Socrates is a man ⊢_{MP} Socrates is mortal
arg(name, beliefs([human(Socrates)], [clause(mortal(X), human(X)])), inf(MP), conclusion([mortal(Socrates)])).
```
Argumentation System

**Argument** is a minimal set of facts that leads to a conclusion through a sequence of inferences

- premises \( \vdash \) conclusion

**Attacks** (or defeat) among arguments are

- **Rebuttal** (\( \leftrightarrow \))
  - premise1 \( \vdash \) conclusion1
  - premise2 \( \vdash \) (not) conclusion1

- **Undercut** (\( \rightarrow \))
  - premise2 \( \vdash \) (not) premise1

**Acceptability** of an argument follows the notion form Dung framework [Dung, 1995]
Argumentation-based Dialogue System

**Argumentation-based dialogue system** is composed of:
- a communication language
- a dialogue protocol
- (a protocol semantics)
Our *communication language* is a set of locutions $L_c$.

### Definition (Locution)

A locution $l \in L_c$ is a expression of the form

$$\text{perf}_{\text{name}}(\text{Arg}_1, \ldots, \text{Arg}_n)$$

where $\text{perf}_{\text{name}}$ is a element of the set $P$ of performatives and $\text{Arg}_x$ is either a fact or an argument.

An agent can perform a locution composed of

- *facts* with syntax $\text{fact}(\text{Terms})$
- *arguments* with syntax $\text{argument}(B, I, C)$

### Information seeking dialogue

Set of moves, identified as “locutions”

OpenDialog, Ask, Tell, DontTell, Provide, Argue
In our framework the dialogue protocol is
- a complete description of all dialogue paths
- a step by step description of the mediator behaviour

Technically the protocol is formalised using process algebra operator (., +, ||, |) respectively sequence, parallel and choice.

Definition (Action)
An action $A$ has the syntax $A := s : L_c | s[t_1, \ldots, t_n] : L_c$ where $s$ indicates the source, and $[t_1, \ldots, t_n]$ indicates the (optional) targets

Definition (Term Action)
A term action $K$ has the syntax
$K := commit(C, X) | read(C, X) | conflict(C, X) | attack(C, X) | defeat(C, X) | acceptS(C, X) | acceptable(C, X)$, where $C$ represents commitment store identifier, and $X$ represents the commitment
Protocol Semantics

**Protocol semantic** is expressed with an operational semantic considering Charles Hamblin’s notion of commitment stores [Hamblin, 1970]

**Definition (operational semantics)**

The *operational semantics* is described by a **Labelled Transition System** \(\langle S, \rightarrow, I \rangle\), where

- \(S ::= (C)P\) represents the state of dialogue system (protocol \(P\) running with commitment store \(C\))
- \(I\) is the set of interactions (labels) composed of \(i ::= \tau | a\)
- \(\rightarrow\) is a transition relation of the kind \(\rightarrow \subseteq S \times I \times S\)

\(s \xrightarrow{i} s'\) in place of \(\langle s, i, s' \rangle \in \rightarrow\) means the dialogue system moves from state \(s\) to \(s'\) due to

- an action \(a\)
- an internal step \(\tau\) (operation over commitment store)
Operational rules that describe the behaviour of term action:

\[
\begin{align*}
(C)commit(x).P & \xrightarrow{\tau} (C'|x)P \\
(C|x)read(y).P & \xrightarrow{\tau} (C|x)P\{x/y\} \\
(C|x)remove(y).P & \xrightarrow{\tau} (C)P\{x/y\} \\
(C|x)conflict(y)).P & \xrightarrow{\tau} (C|x)P \text{ if } \{x \text{ rebuttal } y\} \\
(C|x)attack(y)).P & \xrightarrow{\tau} (C|x)P \text{ if } \{y \text{ undercut } x\} \\
(C|x)defeat(y)).P & \xrightarrow{\tau} (C|x)P \text{ if } \{x \text{ undercut } y\} \\
(C|\mathcal{E})acceptS(y).P & \xrightarrow{\tau} (C|\mathcal{E})P \text{ if } \{\forall E \in \mathcal{E}, y \in E\} \\
(C|E)acceptable(y).P & \xrightarrow{\tau} (C|E)P \text{ if } \{y \in E\}
\end{align*}
\]

(connection to argumentation artifact operations)
In **persuasion dialogue** the goal of a participant is to prove his/her thesis and to rationally persuade the other parties.

- [Walton and Krabbe, 1995] observe that disputes resolution is a subtype of persuasion dialogue

The locutions for persuasion dialogue are:

- **claim** $\varphi$ (assert): The agent asserts a formula $\varphi$ to start the persuasion
- **why** $\varphi$ (challenge): The agent asks for reasons about the $\varphi$ formula
- **concede** $\varphi$ (accept): The agent accepts the validity of $\varphi$
- **reject** $\varphi$ (retract): The agent does not commit the $\varphi$: In some cases it retracts the formula from the commitment store previously stored
- **S since** $\varphi$ (argue): The agent provides reasons for $\varphi$ formula by an argument
Protocol for Persuasion (without CS interaction)

Agent can accept or reject an assertion P based on an internal evaluation of facts and argument acceptability

dialog_persuasion(X,Y,P) :-
    X: assert (argument(true,I,P)).
    dialog_response(X,Y,argument(true,I,P))

dialog_response(X,Y,argument(true,I,P)) :-
    Y: accept (argument(true,I,P)) +
    Y: reject (argument(true,I,P)) +
    Y: why (argument(true,I,P)).
    X: argue (argument(B,I1,P)).
    dialog_argue(X,Y, argue(argument(B,I1,P))).

% Evaluation of chain argument support of P assertion
...
Protocol for Persuasion (with CS interaction) I

\[\text{dialog} \_ \text{argue}(X,Y,\text{argument}(B,I,P)) := \]
\[Y: \text{accept}(\text{argument}(B,I,P)) \cdot \text{commit}(\text{argument}(B,I,P)) + \]
\[Y: \text{reject}(\text{argument}(B,I,P)) + \]
\[Y: \text{argue}(\text{argument}(B1,I1,P1)) \cdot \text{commit}(\text{argument}(B1,I1,P1)) \cdot (\]
\[\text{acceptable}(\text{argument}(B1,I1,P1)) \cdot (\]
\[X: \text{retract}(\text{argument}(B,I,P)) + \]
\[X: \text{argue}(\text{argument}(B2,I2,P2)) \cdot \text{commit}(\text{argument}(B2,I2,P2)) \cdot (\]
\[\text{acceptable}(\text{argument}(B2,I2,P2)) . \]
\[\text{dialog} \_ \text{argue}(X,Y,\text{argument}(B,I,P)) + \]
\[\text{not}(\text{acceptable}(\text{argument}(B2,I2,P2))). \]
\[X: \text{retract}(\text{argument}(B,I,P)) \]
\[) + \]
\[\text{not}(\text{acceptable}(\text{argument}(B1,I1,P1))). \]
\[Y: \text{accept}(\text{argument}(B,I,P)) \cdot \text{commit}(\text{argument}(B,I,P)) \]
\]
Protocol for Persuasion (with CS interaction) II

DA automatically drives the sequence of action through the state of the commitment store using the term actions: commit and acceptable.

- In the choice points some locutions are automatically chosen by preconditions based on the state of acceptability of arguments.

Example

- The proponent agent (X) is constrained to retract the proposal if its supporting argument is not acceptable during the arguing phases.
- The opposer (Y) is constrained to accept the proposal if its opposing argument is not acceptable with respect to the state of the commitment store.
Technological support

Technological support to realize the DA and CAA can be provided by TuCSoN, a coordination infrastructure for MAS.

TuCSoN provides *programmable tuple spaces* where the agents can read/write and consume logic tuples.

**Argumentation process is composed of**

- knowledge representation
- computation over argument sets

**TuCSoN infrastructure supports**

- knowledge declaratively represented in term of logic-tuple arguments
- computation over argument set in term of ReSpecT specification tuples
Example of Run I

Initial dialogue state

dialogstate(persuasion, [act(X, assert1(P)),
    (act(Y, accept(P)) + act(Y, reject(P))) + act(Y, assert1(non(P))) +
    act(Y, why(P)), act(X, argue(argument(N, bel(B), inf(I), conc(C)))) +
    (act(Y, accept(N)) + act(Y, reject(N)))]).

*Olga* asks the possible admissible next locutions by
rd(nextlocutions(persuasion, L)), and the tuple centre responds by
new tuple nextlocution:

nextlocution(persuasion,
    [act(_2, accept(safe)), act(_2, reject(safe)),
    act(_1, assert1(non(safe))), act(_0, why(safe))])
Example of Run II

Figure: (above) **Inspector Tool.** We show the state of the tuple centre after Olga locution by the **inspector tool**

Figure: (below) **CLI Agent.** We start the simulation sending a assert locution in tuple centre from agent Paul by the **CLI Agent**
Conclusions

- We propose a unified framework among dialectical/dialogue system and argumentative reasoning system.
- We propose a more complete formalization of the relation between DA and CAA.
- We propose a model and an infrastructure to realize Alternative Dispute Resolution system in an agent society that merges concepts from argumentation and artifact theories.
  - DA and CAA are a framework to make argumentation and dialectical agent interaction to an operative level.
Bibliography I


Bibliography II


Bibliography III


Arguments and Artifacts for Dispute Resolution

Enrico Oliva    Mirko Viroli    Andrea Omicini

ALMA MATER STUDIORUM—Università di Bologna, Cesena, Italy

WOA 2008 – Palermo, Italy, 18th November 2008