On the Use of Computational Argumentation for Real World Applications

Pavlos Moraitis
LIPADE
Paris Descartes University

Seminar@SystemX
24 JANVIER 2019
Overview

- Introduction
- Basics on Argumentation
- CAFs and application to autonomous driving
- LPP-GORGIAS and some applications
- Conclusions
Why Argumentation?
Human like Systems

- **Natural Intelligence** or high-level cognition is manifested by its handling of **Conflicting Information**

- Argumentation is **native** to human reasoning
  - Role of argumentation in natural human reasoning and dialogue studied in philosophy, linguistics, psychology, ...

- Knowledge captured as arguments

- **Aristotle:** “Dialectic Argument” for handling conflicting positions/claims

Pavlos Moraitis
Why Argumentation?
Logical Reasoning

• **Formal Logic** in terms of Argumentation
  – Argumentation unifies strict/formal and informal reasoning

• **Argumentation** is the primary notion of reasoning.
What is Computational Argumentation?

• **Argumentation** can be abstractly defined as the formal interaction of different conflicting arguments for and against some claim
  – arguments = proofs of claims in some underlying logic
  – The claims may represent beliefs, goals and actions

• **Argumentation process**
  – Construction of arguments (based on different underlying monotonic logics)
  – Definition of interactions between arguments (based on different notions of conflicts)
  – Evaluation of strength of arguments (by using preferences, values, etc.)
  – Definition of status of arguments (i.e. accepted, rejected, undecided based on different acceptability semantics)
  – Choice of winning arguments
What is an argument?

- A set of premises supporting a conclusion / claim

  Information INFO about Paul should be published

  because

  Paul has political responsibilities

  and

  INFO is in the national interest

  and

  if a person has political responsibilities and info about that person is in the national interest then that info should be published
So argumentation is...

• The process whereby arguments are constructed, (possibly) exchanged and evaluated in light of their interactions with other arguments.

- A1: (publish info about Paul because he has political responsibilities )

- A2: (Paul does not have political responsibilities because he resigned)

- A3 : (Paul has still political responsibilities because his resignation has not been accepted)
Argumentation Logics

– **Argumentation logics** formalise defeasible reasoning as construction and comparison of arguments

– Use of monotonic logics for modeling **non-monotonic reasoning** based on the interaction of arguments in the presence of **uncertain**, **incomplete** and **conflicting** knowledge/information
Abstract Argumentation Frameworks

• Structure of arguments is not specified

• Semantics help us to choose the « good/winning » arguments

• Dung’s [Dung 95] acceptability semantics for abstract argumentation frameworks
Dung’s Abstract Argumentation Framework

• $\text{AF} = \langle \text{Args, Attack} \rangle$ where
  – $\text{Args} = \{a_1, \ldots, a_n\}$ is a set of arguments
  – $\text{Attack} \subseteq \text{A} \times \text{A}$ is a binary attacking relation

• $(\text{Args, Attack})$ abstracts from underlying logic based definition of $\text{Args}$ and $\text{Attack}$

• Application of semantics allows us determine winning arguments

\begin{itemize}
  \item A1 (publish)
  \item A2 (not political)
  \item A3 (political)
\end{itemize}
Extension-based Semantics

• **Defense**: counter-attacking of all received attacks

• *(Admissible) Extension* $\mathcal{E}$: conflict-free set of arguments that defends all its members

• **Skeptical and credulous acceptability**

![Diagram]

$\mathcal{E} = \{A_1, A_3\}$
Extension-based Semantics

• **Defense**: counter-attacking of all received attacks

• *(Admissible) Extension* \( E \): conflict-free set of arguments that defends all its members

• **Skeptical and credulous acceptability**

\[ \checkmark \quad \checkmark \quad \times \quad \checkmark \]

\[ \{A_1, A_3\}, \quad \{A_2\} \]

\[ \varepsilon_1 = \{A_1, A_3\}, \quad \varepsilon_2 = \{A_2\} \]
Extension-based Semantics

• **Defense**: counter-attacking of all received attacks

• *(Admissible)* Extension $\mathcal{E}$: conflict-free set of arguments that defends all its members

• **Skeptical and credulous acceptability**

\[ \mathcal{E} = \{A_1, A_3\} \]

\[ \mathcal{E}_1 = \{A_1, A_3\}, \mathcal{E}_2 = \{A_2\} \]

\[ \mathcal{E}_1 = \{A_1, A_3\}, \mathcal{E}_2 = \{A_1, A_2\} \]
Dynamics in Argumentation

• Addition/Removal of Arguments

• Addition/Removal of Attacks

• Goal: obtaining the acceptance of a particular (set of) argument(s)
Control Argumentation Frameworks (CAFs)

[Dimopoulos, Mailly, Moraitis (AAAI18)]

- A **CAF** is an argumentation framework where arguments are divided in three parts: **fixed**, **uncertain** and **control**

- **Fixed**: background knowledge about a static environment

- **Uncertain**: changes that may occur in the environment

- **Control**: possible remedial actions of the agent against possible negative effects of changes
Control Argumentation Frameworks (CAFs)

• Implementation of self-adaptive systems ensuring real time control tasks in different contexts such as:
  – autonomous driving
  – smart homes
  – surveillance of buildings and streets
  – personalized self-regulation services for humans
  – recommendation policies in finance
  – risk management
  – etc.
Control Argumentation Frameworks (CAFs)

CAF Theory

\( F: < A_F = \{a,b,d,g\}, \overset{\rightarrow}{\longrightarrow} =\{(a,b),(b,a),(c,a)\}> \)

\( U: < A_U = \{c\}, \overset{\leftrightarrow}{\leftrightarrow} =\{(c,g),(g,c)\}, \overset{\rightarrow}{\longrightarrow} =\{(d,a)\}> \)

\( C: < A_C = \{e,f\}, \overset{\rightarrow}{\longrightarrow} =\{(e,c),(f,d)\}> \)
Control Argumentation Frameworks (CAFs)

Dynamics Under Uncertainty + Computational Methods

Completions
Control Argumentation Frameworks (CAFs)

Dynamics Under Uncertainty + Computational Methods

Completions
Control Argumentation Frameworks (CAFs)

Dynamics Under Uncertainty + Computational Methods

Completions

Controllable CAFs

Given a target \( T \subseteq A_F \) and a semantics \( \sigma \), \( \text{CAF} \) is skeptically (resp. credulously) controllable w.r.t. \( T \) and \( \sigma \) if \( \exists A_{\text{conf}} \subseteq A_C \) s.t.:

- \( \text{CAF}' \) is the result of configuring \( \text{CAF} \) by \( A_{\text{conf}} \)
- \( T \) is included in every (resp. at least one) \( \sigma \)-extension of every completion of \( \text{CAF}' \)

\( T=\{a\} \)
**Control Argumentation Frameworks (CAFs)**

Dynamics Under Uncertainty + Computational Methods

**Completions**

**Controllable CAFs**

Given a target $T \subseteq A_F$ and a semantics $\sigma$

*CAF* is skeptically (resp. credulously) controllable w.r.t. $T$ and $\sigma$ if $\exists A_{conf} \subseteq A_C$ s.t.:

- $CAF'$ is the result of configuring $CAF$ by $A_{conf}$
- $T$ is included in every (resp. at least one) $\sigma$-extension of every completion of $CAF'$

Use of a QBF-based method

$T=\{a\}$  \  $\mathcal{E}=\{a,e,f,g\}$

Pavlos Moraitis
CAFs Application to Autonomous Driving
mf: move forward
mb: move backward
tl: turn left
tr: turn right
st: stop
covr: car overtakes right
covl: car overtakes left
ovl: overtake left
obf: obstacle in front
br: brake
ac: accelerate
av: avoid
sd: slow down

Pavlos Moraitis
\[ T = \{ mf \} \]

---

**Main [Java Application]**

- `lnbr`: 0
- `subsumption`: 0
- `preprocessor`: 1
- `variable elimination`: 1
- `upla`: 1
- `solve`: 1
- `maxmode`: 1
- `solvemode`: QBF
- `cpu time (loading CNF)`: 0.002s

**Statistics**

- `#liters`: 0
- `#clauses`: 0
- `#decisions`: 0
- `#BCP operations`: 163
- `#confs`: 0
- `#sol`: 1
- `#pureLiterals`: 0
- `#dcLiterals`: 0
- `#restarts`: 0
- `#reconf`: 0
- `#clsimplifications`: 0
- `#cusimplifications`: 0
- `#lhnbr`: 0
- `#lnbr`: 0
- `#inpro`: 0
- `cpu time`: 0.003s

**Result**

- SATISFIABLE
- **Control Configuration**: `c_arg(avoid). c_arg(brake). c_arg(slowDown)`
- **Duration**: 270ms

---

**Main [Java Application]**

- `lnbr`: 0
- `subsumption`: 0
- `preprocessor`: 1
- `variable elimination`: 1
- `upla`: 1
- `solve`: 1
- `maxmode`: 1
- `solvemode`: QBF
- `cpu time (loading CNF)`: 0.002s

**Statistics**

- `#liters`: 0
- `#clauses`: 0
- `#decisions`: 0
- `#BCP operations`: 163
- `#confs`: 0
- `#sol`: 1
- `#pureLiterals`: 0
- `#dcLiterals`: 0
- `#restarts`: 0
- `#reconf`: 0
- `#clsimplifications`: 0
- `#cusimplifications`: 0
- `#lhnbr`: 0
- `#lnbr`: 0
- `#inpro`: 0
- `cpu time`: 0.003s

**Result**

- SATISFIABLE
- **Control Configuration**: `c_arg(avoid). c_arg(accelerate). c_arg(slowDown)`
- **Duration**: 193ms
Structured Argumentation Frameworks
Logic Programming with Priorities (LPP)  
[Kakas&Moraitis, (AAMAS03)]

• Logic Programming without Negation as Failure (LPwNF) (Kakas, Mancarella, Dung, ICLP94; Dimopoulos&Kakas, ILPS95)
  – In the LPwNF logic programs are non-monotonic theories  
  – Each logic program is viewed as pool of default sentences from which we must select a suitable subset, called extension to reason with  
  – Sentences in a logic program are written in the usual logic programming language with the addition of an explicit negation but without the NAF operator (i.e. not)
Logic Programming with Priorities

• A theory is a pair (T,P) whose sentences are formulae, in a background monotonic logic (L, ⊢), of the form \( L \leftarrow L_1, \ldots, L_n \), where \( L, L_1, \ldots, L_n \) are positive or negative ground literals

• For rules in P the head \( L \) refers to an (irreflexive) higher-priority relation. \( L \) has the general form \( L = h-p(rule_1, rule_2) \) where \( rule_1 \) and \( rule_2 \) are unique names of rules in the theory

• The derivability relation, ⊢, of the background logic is given by the single inference rule of modus ponens
Logic Programming with Priorities

An LPP theory $T$ is a tuple $T=(\mathcal{I}, \mathcal{P})$ where:

- $\mathcal{I}$ is a set of object level arguments supporting a set of options $O$
- $\mathcal{P}$ is a set of priority arguments that is partitioned into a finite set of levels, $\mathcal{P}=(\mathcal{P}_1, \ldots, \mathcal{P}_n)$
- All the arguments in $\mathcal{P}_1$ are priority arguments $p^1_{12}(\text{arg}_1 > \text{arg}_2)$, supporting preferences between arguments $\text{arg}_1, \text{arg}_2 \in \mathcal{I}$
- For any $1 < k \leq n$, all arguments in $\mathcal{P}_k$ are priority arguments, $p^k_{12}(\text{q}_1 > \text{q}_2)$, supporting a preference between $\text{q}_1, \text{q}_2 \in \mathcal{P}_{k-1}$
Logic Programming with Priorities

- **Object level rules:**
  - \( r_i : L \leftarrow L_1, \ldots, L_n \)
  - \( r_j : \neg L \leftarrow L_1, \ldots, L_m \)

- **Higher Priority rules:**
  - \( p^1_{ij} : \text{h-p}(r_i, r_j) \leftarrow \text{true (i.e. generally) (or conditions}_{ij} \)
  - \( p^1_{ji} : \text{h-p}(r_j, r_i) \leftarrow \text{conditions}_{ji} \)
  - \( p^2_{ji} : \text{h-p}(p^1_{ji}, p^1_{ij}) \leftarrow \text{true (or conditions}_{ji} \)
  - \( p^2_{ij} : \text{h-p}(p^1_{ij}, p^1_{ji}) \leftarrow \text{conditions}_{ij} \)
  - .......

Pavlos Moraitis
Gorgias* Argumentation Technology

• **Principled** Declarative Problem Solving via **Argumentation**

  – Solid *theoretical foundation* for building acceptable arguments:

  • Argument(s) for one option is (are) **strong** enough to **defend** against all its (their) **counter-arguments** for other options

Policy Compliance ⇔ Acceptable Arguments

*Greek Sophist c.485 — c.380 BCE*
Policy Decisions: Challenges

• **Policy Compliance** of systems, robust under incomplete & inconsistent information in:
  – Dynamic environments
    • Internal conflicts inside a policy
  – Multi-policy environments
    • External conflicts across policies

• **Flexibility** in Development of systems
  – Modular Adaptation to changes in policies
  – Accommodate new policies
Policy Decisions: Challenges

• **Explainability of Systems**

  – New EU law to give everyone a **right to an explanation** of any decision affecting them that has been **reached algorithmically**.

  – **Explain** level of access granted.
The Gorgias System (2004 -...)

- Builds **sound** preferred acceptable arguments from expert/policy knowledge.

- Realizes **Decision Making** through argumentation for **application** problems

- **Flexible and Robust** system
  - Incomplete, **contextual** and **conflicting** knowledge
  - Consideration of **different** (conflicting) view points

- **Scenario-based** knowledge engineering

- **Real-life** applications since 2004
Gorgias Application Approach

• **Knowledge** as *Argument Schemes* via *Scenarios*

• **Knowledge acquired by:**
  – Elicited from Experts
  – Machine Learned
  – Hybrid Acquisition

• **Knowledge types:**
  – Expert
  – Common Sense
  – Personal biases
Real-life Applications of Gorgias


- 2017: MEDICA – Regulate Data Access
  - DEMO: Automating Legislation for access to patient data

- 2017: Data Share Agreements (for health data)
  - Coco Cloud: EU project at Imperial College.

- 2018: Cyber attack management...
Medical Data Access/Sharing

• **Problem:** Decide **Level of Access** according to user and current circumstances

• There are **6 Access Levels** (Read & Write)
  – Full Access
  – Partial Access
  – Read Only Access
  – Restricted Read Access
  – Suspended Access
  – No Access

Law **138(I)/2001**: Personal Data Protection

Law **N. 1(I)/2005**: Patient Rights
Medical Data Access: MEDICA

- MEDICA: http://medica.cs.ucy.ac.cy
- Demo Online
- Pilot evaluation
## Medical Data Access

<table>
<thead>
<tr>
<th>Scenarios \ Options</th>
<th>( O_2 = \text{deny_access} )</th>
<th>( O_1 = \text{allow_access} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S_1 = {\text{sick, privateData}} )</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>( S_2 = {\text{hospitalDoctor, privateData}} )</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>( S_{12}^1 = {\text{hospitalDoctor, privateData, sick}} \cup {\text{hospitalized}} )</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>( S_{12}^2 = {\text{hospitalized, hospitalDoctor, privateData, sick}} \cup {\text{unconscious}} )</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>( S_{12}^3 = {\text{emergency, hospitalized, hospitalDoctor, privateData, unconscious}} \cup {\text{permission}} )</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
Decision policy of the agent

\( r_1: \text{allowAccess} \leftarrow \text{sick, privateData} \)

\( r_2: \text{denyAccess} \leftarrow \text{hospitalDoctor, privateData} \)

\( p_{121}^1: h-p(r_2, r_1) \leftarrow \text{true} \)

\( p_{12}^1: h-p(r_1, r_2) \leftarrow \text{hospitalized} \)

\( p_{212}^1: h-p(p_{12}^1, p_{21}^1) \leftarrow \text{true} \)

\( p_{221}^1: h-p(p_{21}^1, p_{12}^1) \leftarrow \text{unconscious} \)

\( p_{21}^2: h-p(p_{221}^1, p_{12}^1) \leftarrow \text{true} \)

\( p_{12}^2: h-p(p_{21}^2, p_{221}^1) \leftarrow \text{permission} \)

\( p_{12}^3: h-p(p_{12}^2, p_{21}^2) \leftarrow \text{true} \)
Eye Clinic Cognitive Assistant

• Provides a **first level support** to patients at the reception of the clinic:
  
  – Finds **most expertly probable** diseases
  – Able to recognize the possibility of **severe/urgent** diseases
  – Suggests **extra information/tests** needed to **focus** on the **probable** disease.
Eye Clinic Cognitive Assistant

• Human-like interaction with patients and/or nurse receptionist:
  – Input: Symptoms & test results of patient in their natural form.

  – Output:
    • Naturally presented probable disease(s), urgency level and further tests when needed
    • Non-technical explanation of diagnosis
Eye Clinic Cognitive Assistant

- Scale of **full** expert knowledge:
  - 80 diseases
  - Many Hundreds of scenarios
  - 35+ Parameters (symptoms, zones, contexts)
  - 3-4 Levels of hierarchy of scenarios
  - Several Hypotheticals
## Eye-Clinic System Scenarios

<table>
<thead>
<tr>
<th>Scenarios \ Diseases</th>
<th>$D_i$</th>
<th>$D_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_i$ = {s(1), s(2), s(3)}</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>$S_j$ = {s(1), s(3), s(4)}</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>$S_{ij}^1$ = {s(1), s(2), s(3), s(4)}</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>$S_{ji}^1$ = {s(1), s(2), s(3), s(4)}</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>$S_{ji}^2$ = {s(1), s(2), s(3), s(4)} $\cup$ {s(8)}</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>$S_{ij}^2$ = {s(1), s(2), s(3), s(4)} $\cup$ {s(19)}</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>$S_{ij}^3$ = {s(1), s(2), s(3), s(4), s(8), s(19)}</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
Eye-Clinic System (example)
Assuming that qa(atcd_uaa)
Call Assistant

• Requirements
Paul wants to train his personal assistant to manage his calls. He wants him to do one out of two options. The first is to allow the phone to ring when there is a call, the second is to deny the call. In general he chooses the first option over the second. If he is at work, however, that is a reason to deny the call. When he is at work he prefers to allow family calls over denying them, except when he is in a meeting, when he prefers to deny over allowing the call. Being in a meeting there is a possibility to prefer to accept a call from his son when he is at school. He will accept it if he believes that his son is ill.
Call Assistant Scenarios

<table>
<thead>
<tr>
<th>Scenarios \ Options</th>
<th>$O_1$=allow(Call)</th>
<th>$O_2$=deny(Call)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S^0=${phone_call}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$S^1_{12}$={phone_call}</td>
<td>$X$</td>
<td>$X$</td>
</tr>
<tr>
<td>$S^1_{21}=${phone_call} $\cup$ {at_work}</td>
<td></td>
<td>$X$</td>
</tr>
<tr>
<td>$S^2_{12}=${phone_call, at_work} $\cup$ {familly_member(Call)}</td>
<td></td>
<td>$X$</td>
</tr>
<tr>
<td>$S^3_{21}=${phone_call, at_work, familly_member(Call)} $\cup$ {in_meeting}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$S^3_{12}=${phone_call, at_work, familly_member(Call), in_meeting} $\cup$ {from_son(Call), son_at_school, son_is_ill}}</td>
<td>$X$</td>
<td></td>
</tr>
</tbody>
</table>
Decision policy of the call assistant agent

\[r_1: \text{allow}(\text{Call}) \iff \text{phone\_call}\]
\[r_2: \text{deny}(\text{Call}) \iff \text{phone\_call}\]
\[p_{12}^1: h-p(r_1, r_2) \iff \text{true}\]
\[p_{21}^1: h-p(r_2, r_1) \iff \text{at\_work}\]
\[p_{21}^2: h-p(p_{12}^1, p_{12}^1) \iff \text{true}\]
\[p_{12}^2: h-p(p_{12}^1, p_{21}^1) \iff \text{family\_member}(\text{Call})\]
\[p_{12}^3: h-p(p_{12}^2, p_{21}^2) \iff \text{true}\]
\[p_{21}^3: h-p(p_{21}^2, p_{21}^2) \iff \text{at\_meeting}\]
\[p_{21}^4: h-p(p_{21}^3, p_{21}^3) \iff \text{true}\]
\[p_{12}^4: h-p(p_{12}^3, p_{12}^3) \iff \text{from\_son}(\text{Call}), \text{son\_at\_school}, \text{son\_is\_ill}\]
\[p_{12}^5: h-p(p_{12}^4, p_{21}^4) \iff \text{true}\]
Pavlos Moraitis

Gorgias-B, File: C:\Users\Pavlos Moraitis\Desktop\Gorgias_B_last\Gorgias_B\Gorgias_B.exe

Options View
Option predicate: ( )

Options
allow(Call)
deny(Call)

Incompatible Options
allow(Call) deny(Call)
deny(Call) is complement to allow(Call)
allow(Call) is complement to deny(Call)

Arguments View - Argue at 1st level
Select option
allow(Call)

Condition
in general choose allow(Call)

When [phone_call] choose allow(Call)
When [phone_call] choose deny(Call)

Argue at higher levels
Select level of arguing 2

Scenario with conflicting options
1: phone_call

Select predicate  phone_call/0
Edit parameters
Add predicate
Defined models based on the selected scenario
When [at_work, phone_call] prefer deny(Call) over allow(Call)
When [phone_call] prefer allow(Call) over deny(Call)

Argue at higher levels
Select level of arguing 3

Scenario with conflicting options
1: at_work, phone_call

Select predicate  at_work, phone_call
Edit parameters
Add predicate
Defined models based on the selected scenario
When [at_work, phone_call] prefer deny(Call) over allow(Call)
When [at_work, family_member(Call), phone_call] prefer allow(Call) over deny(Call)

Return to simple scenarios
Resolve Conflicts
Pavlos Moraitis

Run scenarios
Instantiate the scenario knowledge for querying
at_meeting/0  
allow  
Add fact

Model instantiation monitor

---- New goal: Explore all options! ----
- Instantiated facts:
  phone_call
  at_work
  family_member(call1)
  at_meeting

- New goal: allow(Call)?

No solution for this goal.
- New goal: deny(Call)?

Found solution:
Variable Call instance: call1
Argument #1:
When [phone_call] choose deny(call1)
When [at_meeting, at_work, phone_call, family_member(call1)]
When [at_work, phone_call] prefer deny(call1) over allow(call1)

Run scenarios
Instantiate the scenario knowledge for querying
son_is_ill/0  
allow  
Add fact

Model instantiation monitor

- Instantiated facts:
  phone_call
  at_work
  family_member(call1)
  at_meeting
  from_son(call1)
  son_at_school
  son_is_ill

- New goal: allow(Call)?

Found solution:
Variable Call instance: call1
Argument #1:
When [phone_call] choose allow(call1)
When [phone_call] prefer allow(call1) over deny(call1)
When [son_at_school, at_meeting, son_is_ill, from_son(call1), at_work, family_member(call1), phone_call]
When [at_work, family_member(call1), phone_call] prefer allow(call1) over deny(call1)

- New goal: deny(Call)?

No solution for this goal.
Argumentation-based Automated Negotiation

• Conflict resolution concerning a specific issue related to a resource sharing (e.g. the price of a product)
• Agents exchange offers supported by arguments
• Search for an agreement through the exchanged arguments
• Proponents (agents) defend the supporting arguments by attacking the opponents arguments that attack them, etc.
Negotiation with CAFs
[Dimopoulos, Mailly, Moraitis (AAMAS19)]

Initial theories of agents $\alpha$ and $\beta$: each agent uses a CAF for representing the incomplete knowledge he has about the profile of his opponent.
The goal of agent $\beta$ is to persuade agent $\alpha$ to accept the supporting argument $X$ (and therefore offer $O$) by defending argument $X$ with the control arguments $D$ and $F$.
The Negotiation Dialogue between Agent $\alpha$ and Agent $\beta$
Conclusions

• Computational argumentation is now mature enough for real world applications

• CAFs are very well suited for modeling self-adaptive systems

• LPP and GORGIAS are very well suited for modeling decision policies under incomplete, contextual and conflicting knowledge
References

• Related to the application of LPP-GORGIAS and CAFs in single agent reasoning

  - Dimopoulos Y., Mailly JG., Moraitis P., "Control Argumentation Frameworks", in 32nd AAAI Conference on Artificial Intelligence (AAAI'18), pp. 4678-4685, New Orleans, USA, 2018
References (cont.)

• Related to the application of LPP-GORGIAS and CAFs in multi-agent systems (i.e. agent dialogues)

