Formal Argumentation

Bruno Yun

Dept. of Computing Science, University of Aberdeen, UK

22nd European Agent Systems Summer School
Outline

1. Introduction
2. Abstract Argumentation
3. Proof Dialogues
4. Ranking-based Semantics
5. Structured Argumentation
6. Applications
Outline

1. Introduction
2. Abstract Argumentation
3. Proof Dialogues
4. Ranking-based Semantics
5. Structured Argumentation
6. Applications
Recall that agent reasoning is often implemented using a rule based system, for both practical and epistemic reasoning.

E.g., using Modus Ponens

\[
\text{All men are mortal} \\
\text{Socrates is a man} \\
\hline
\text{Therefore Socrates is Mortal}
\]

\[
\text{man} \rightarrow \text{mortal}, \ socrates \rightarrow \text{man} \text{ therefore, given socrates we obtain mortal}
\]
The real world is a bit more complicated...

- Imagine instead the following rules.
  
  \[ t \rightarrow b : \text{Tweety is a bird} \]
  
  \[ b \rightarrow f : \text{Birds fly} \]

- We can deduce that Tweety flies.
  
  \[ b \rightarrow f \quad t \rightarrow b \text{ yields } f \]

- But what about

  \[ \text{Birds fly } b \rightarrow f \]
  
  \[ \text{Penguins are birds } p \rightarrow b \]
  
  \[ \text{Penguins don’t fly } b \rightarrow \neg f \]
  
  \[ \text{Tweety is a penguin } t \rightarrow p \]

- Now we obtain \( f \) and \( \neg f \), contradiction.
This is an example of **non-monotonic** reasoning.

- Adding more rules causes certain conclusions to no longer be valid.
- Many logics (and therefore reasoning procedures) can’t handle such non-monotonicity.
- We’re going to look at a simple, easily understood approach to non-monotonic reasoning.
- But first, we need to take a high level view of things.
Some psychologists claim that the ability to reason with arguments is fundamental to human intelligence, and indeed that human intelligence evolved so as to allow us to reason with arguments.

Formal argumentation theory is the study of reasoning with arguments.

We can consider **abstract** and **structured**, or instantiated argumentation

- Abstract argumentation considers how arguments interact
- Structured argumentation considers how arguments are formed
Outline

1. Introduction
2. Abstract Argumentation
3. Proof Dialogues
4. Ranking-based Semantics
5. Structured Argumentation
6. Applications
Abstract argumentation

- We begin by considering Dung’s 1995 seminal abstract argumentation framework.
- We do not care about what an argument is, rather how it interacts with other arguments.
- Arguments interact by **defeating** each other.
- E.g., if I argue its cloudy \((c)\), and you argue its sunny \((s)\), these two arguments defeat each other.
- So an argumentation system consists of a set of arguments \(\mathcal{A}\), and a binary defeat relation \(\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}\).
- We say that \(a\) defeats \(b\) if \((a, b) \in \mathcal{R}\)

\[
AF = (\{c, s\}, \{(c, s), (s, c)\})
\]

- Can the defeat relation be non-symmetric?
Can the defeat relation be non-symmetric?

Yes

- $r$: This object looks red, and — given that an object looks red it is red — this object is red
- $l$: Objects under a red light might look red but are not, and this object is under a red light

\[ l \rightarrow r \]
We can represent an argument framework as a tuple $AF = (A, R)$, but we can just as easily encode them as a graph.

$$AF = (\{c, s\}, \{(c, s), (s, c)\})$$
Argument interactions

- We know how to construct an argument framework (AF).
- But — given such an AF, which arguments are justified?
- That is, what would a “reasonable” or “rational” reasoner conclude?
- Let’s start with some examples.
Example 1

- $t$: "Tweety is a bird", (or equivalently $t$: "Tweety is a bird, birds fly, therefore Tweety flies")
- $AF = (\{t\}, \{\})$
Example 2

- $a$: "Tweety is a dog and therefore she flies"
- $b$: "Dogs do not fly"
- $AF = (\{a, b\}, \{(b, a)\})$
Example 3

- **a**: "Tweety is a dog and therefore she flies"
- **b**: "Dogs do not fly"
- **c**: "On the planet we have travelled to, dogs do fly"
- \( AF = (\{a, b, c\}, \{(b, a), (c, b)\}) \)
Example 3

- $a$: "Tweety is a dog and therefore she flies"
- $b$: "Dogs do not fly"
- $c$: "On the planet we have travelled to, dogs do fly"
- $AF = \{a, b, c\}, \{(b, a), (c, b)\}$

This is an example of **reinstatement**, the presence of $c$ reinstates $a$, making it justified.
Example 4

- $c$: "It’s cloudy"
- $s$: "It’s sunny"
- $AF = (\{c, s\}, \{(c, s), (s, c)\})$
Some terminology

- A set of arguments we consider justified is called an extension.
- Different argumentation semantics can yield different extensions.
Basic properties — Conflict freeness

- A set of arguments $S \subseteq A$ is **conflict free** if and only if there is no $a, b \in S$ such that $(a, b) \in R$.

- Arguably, any rational reasoner would not consider arguments in conflict as justified, and conflict freeness is therefore a basic property required for any extension.
An argument \( a \in A \) is **acceptable** w.r.t. a set of arguments \( S \subseteq A \) if and only if for any argument \( b \) such that \( (b, a) \in R \) there is some \( c \in S \) such that \( (c, b) \in R \).

E.g., \( x \) is acceptable w.r.t \( \{y, z\} \).
Justified (attempt 1)

- A set of arguments $S$ is **justified** if all arguments within $S$ are acceptable with respect to $S$. 
A set of arguments $S$ is **justified** if all arguments within $S$ are acceptable with respect to $S$.

Problem: $\{x, b\}$ is justified
A set of arguments $S$ is **admissible** if
- all arguments within it are conflict free
- and all arguments within it are acceptable w.r.t. $S$

The empty set is always admissible
Is admissibility enough?

- What is admissible here?
Is admissibility enough?

- What is admissible here?

- Admissibility gives us reinstatement, but the empty set is also admissible.

- So we’re on the right track.
An admissible set is a **complete extension** if it includes all arguments made acceptable by the set.

Here, only \( \{c, a\} \) is a complete extension, as \( \{\} \) does not include \( c \) which is acceptable.
Complete Extensions

- \{a\}, \{b, d\}, {} are the three complete extensions.
- So what’s going on? Which of these would a rational reasoner agree to?
Skeptical vs Credulous Reasoning

- A **skeptical** reasoner only accepts what is undeniably true.
- A **credulous** reasoner will accept things that are possibly true.

Skeptical reasoner: "I can’t decide and so won’t accept either conclusion as justified". \{\}

Credulous reasoner: "It is either sunny, or cloudy (but clearly not both)". \{{s}, {c}\}
Grounded Extension

- The **grounded** extension is the minimal complete extension (with respect to set inclusion).
- Grounded extensions represent a form of skeptical reasoning.
- The grounded extension is unique.
- The grounded extension is included in any other complete extension.
Another way of viewing the grounded extension is that it includes everything that is unquestionably true (i.e., undefeated).

And everything defended by the above.

And everything defended by that.

... 

Until nothing more can be added.
A preferred extension is a maximal complete extension (with respect to set inclusion).

Preferred extensions represent a form of credulous reasoning.

Multiple preferred extensions can exist.

\{\{s\}, \{c\}\} are the two preferred extensions.
A credulous reasoner might consider any preferred extension as a possibility.

A skeptical reasoner (different to the grounded skeptical reasoner) might consider arguments in the intersection of the preferred extensions as plausible. These are the skeptical preferred semantics.

The grounded extension is more skeptical, i.e., excludes more arguments, than the skeptical preferred semantics.
Stable Extensions

- Stable extensions require everything not considered justified to be explicitly rejected.
- Formally, stable extensions are complete extensions where any argument not present in an extension is defeated by an argument in the extension.

\[ \{a, d\} \text{ and } \{b, d\} \text{ are the stable extensions, and the preferred extensions.} \]

- In fact, any stable extension is a preferred extension.
In fact, any stable extension is a preferred extension.

But not vice-versa
Stable extensions

- In fact, any stable extension is a preferred extension.
- But not vice-versa

\[ \{a\} \text{ and } \{b, d\} \text{ are the preferred extensions.} \]

\[ \{b, d\} \text{ is the only stable extension.} \]
Exercise 1

Questions and answers are online at:

itempool.com/BrunoYun/live
Exercise 2

Questions and answers are online at:

itempool.com/BrunoYun/live
Labelings are another popular way to obtain extensions, and provide some additional semantics.

Labelings are built on the following observations.

- An argument should be considered IN if it is not defeated.
- It should be considered OUT if it is defeated.
- An argument’s status can be UNDEC.
Labelings

- When is an argument defeated (i.e., OUT)?
  - When one of its defeaters is definitely undefeated, i.e., IN.
- When is an argument definitely undefeated (i.e., IN)?
  - When all of its defeaters are definitely defeated, i.e., OUT.
- Otherwise, the argument’s status is uncertain, i.e., UNDEC.
- A labelling of arguments which adheres to these rules is a legal labelling.
- Furthermore, the IN labelled arguments of such a labelling correspond to a complete extension.
Given that we have complete labelings, we can easily generate grounded and preferred labelings — whose IN arguments correspond to grounded and preferred extensions — by minimising or maximising the number of arguments labelled IN.

A link between stable labelings and extensions can similarly be defined by requiring no arguments to be labelled UNDEC.

Note however that labelings convey additional information about argument status when compared to extensions.
We can recognise arguments whose truth value is truly undecided (as opposed to true or false in some worlds).
There is only one legal labelling:

\[
(\{a, c, e\}, \{d, b\}, \emptyset)
\]
Labelings

There are three legal labelings:

1. $\{\{a\}, \{b\}, \{c, d, e\}\}$
There are three legal labelings:

1. \( (\{a\}, \{b\}, \{c, d, e\}) \)
2. \( (\{b, d\}, \{a, c, e\}, \emptyset) \)
3. \( (\emptyset, \emptyset, \{a, b, c, d, e\}) \)

The (1) and (2) are preferred labelings but only (2) is a stable labelling.
We can also introduce new semantics, namely ones where the number of UNDEC is minimal.

There are myriad other semantics (e.g., ideal, CF2) which we do not discuss here.
Determining whether an argument is in the grounded extension can be done in polynomial time.

Determining whether a set of arguments is admissible, conflict free or a stable extension can be done in polynomial time.

Nearly everything else (verifying or finding a preferred extension for example) is computationally difficult in the worst case\(^1\).

\(^1\)See Chapter 5 of "Argumentation in Artificial Intelligence", 2009 for details.
Relationships between semantics

- Stable labelling
  - Semi-stable labelling
    - Preferred labelling
      - Complete labelling
        - Admissible labelling
          - Conflict-free labelling
  - CF2 labelling
  - Ideal labelling
    - Stage labelling

Exercise 3

Questions and answers are online at:

itempool.com/BrunoYun/live
Outline

1. Introduction
2. Abstract Argumentation
3. Proof Dialogues
4. Ranking-based Semantics
5. Structured Argumentation
6. Applications
When arguing, humans engage in dialogue.

Or internally, in monologue.

Such proof dialogues are a natural way of determining whether an argument is justified according to some semantics.
Proof dialogues are usually assumed to take place between two parties
- PRO, who wishes to demonstrate that the argument is justified.
- CON, who wishes to demonstrate the argument is not justified.

Dialogues are often specified in terms of **dialogue games**, with rules describing
- Who may make an utterance in the dialogue at any point in time
- What the nature of the utterance may be
- When the dialogue ends (and begins)
- Who wins the dialogue
Grounded Semantics — Game 1

- Players take turns advancing a single argument, with PRO moving first.
- Following the first turn, a player must advance an argument defeating the previously advanced argument.
- One additional restriction is that PRO may not advance an argument they have advanced before (though CON may).
- The player making the last legal move is the winner of the game.
- Under these rules, there is a winning strategy for PRO if and only if the moves advanced by it are in the grounded extension.
- But if PRO makes a mistake, they can still lose the game.
The winning strategy is $a, b, c, d, e$

But if PRO plays $d$ instead of $c$, then CON will win, either with

- $a, b, d, e$; or
- $a, b, d, c$
While a single iteration of the game has complexity $O(|\mathcal{A}|)$, one could potentially need to explore an exponential number of games to ensure whether an argument is in the grounded semantics. 

Given that the grounded extension can be computed in polynomial time, this suggests that a better dialogue game for the extension does exist.
Grounded Discussion Game (GDG)

- An argument is in the grounded extension if it "has to be the case".
- For an opponent to show an argument is not in the grounded extension, they simply need to show that one of its defeaters "could be the case".
- The burden of proof is thus on PRO to show that none of the defeaters of the argument they are defending can be the case.
- So the moves for the grounded game are
  - $HTB(A)$: $A$ has to be the case — $A$ is in the grounded labelling. Moved by PRO.
  - $CB(B)$: $B$ is not out in the grounded labelling. Moved by CON.
  - $CONCEDE(A)$: Signals an agreement that $A$ is in. Moved by CON.
  - $RETRACT(B)$: Signals that $B$ is out. Moved by CON.
Grounded Discussion Game (GDG)

- Game starts with PRO making a *HTB* statement.
- CON Can then make one or more *CB*, *CONCEDE* and *RETRACT* statements.
- After which PRO makes a *HTB* and the cycle repeats.
- N.B., CON makes multiple moves for every PRO move.
Rules

- $HTB(A)$ is either the first move, or the previous move was $CB(B)$ in which case $A$ must defeat $B$, and CON can’t $CONCEDE$ or $RETRACT$.

- $CB(B)$ is moved when $B$ defeats the last $HTB(A)$ statement where $CONCEDE(A)$ has not yet been made; $B$ has not been retracted; the last move was not a $CB$ move, and $CONCEDE$ and $RETRACT$ cannot be played.

- $CONCEDE(A)$ can be played when $HTB(A)$ was moved earlier, and all defeaters of $A$ have been retracted, and $CONCEDE(A)$ has not been played.

- $RETRACT(A)$ can be played when $CB(A)$ was moved in the past, and an defeaters of $A$ has been conceded, and $RETRACT(A)$ has not been played.
Winning and Losing

- If CON concedes the original argument, PRO wins. Otherwise, CON wins.
- If a HTB-CB repeat occurs for the same argument, CON wins (due to burden of proof).

```
1: PRO : HTB(C)
2: CON : CB(B)
3: PRO : HTB(A)
4: CON : CONCEDE(A)
5: CON : RETRACT(B)
6: CON : CONCEDE(C)
```
Winning and Losing

- If CON concedes the original argument, PRO wins. Otherwise, CON wins.
- If a HTB-CB repeat occurs for the same argument, CON wins (due to burden of proof).

1: PRO : $HTB(B)$
2: CON : $CB(A)$
Winning and Losing

- If CON concedes the original argument, PRO wins. Otherwise, CON wins.
- If a HTB-CB repeat occurs for the same argument, CON wins (due to burden of proof).

```
1: PRO : HTB(F)  4: CON : CONCEDE(A)
2: CON : CB(B)  5: CON : RETRACT(B)
3: PRO : HTB(A)  6: CON : CB(A)
```
GDG vs Game 1

- The first game allows argument to reappear over multiple paths. In worst case, it’s exponential in the number of arguments in the framework.
- GDG considers each argument once, and is linear in the number of arguments in the framework (note that a strategy exists which minimises game length).
- Exponential blow-up is a standard feature of most tree-based discussion games.
Skeptical Preferred Semantics

- Grounded is considered "too skeptical".
- Credulous preferred is "too lenient".
- Skeptical preferred semantics seem to capture human intuitions well.
- Some work uses meta-dialogues, or works only where stable and preferred semantics coincide.
Approach

- Two players, $O$ and $P$
- Two phases
  - Phase 1: $O$ advances an extension where the argument under discussion is out or undec.
  - Phase 2: $P$ shows that this extension is not a preferred extension.
- Under perfect play, $O$ will win iff the focal argument is not in, with $P$ winning otherwise.\(^2\)

More detail

- **Moves:**
  - **What is** (WI) — requests a label to be assigned to an argument.
  - **Claim** (CL) — assign a label to an argument.

- Players take turns to make a single move, with $P$ beginning both phases.

- **Phase 1:**
  - $P$ plays *WI* moves (starting with argument of interest).
  - $O$ responds with a *CL* move assigning a (legal) label to the argument.
  - $P$’s *WI* moves are for arguments which attack a previous *CL* move (and no *CL* for that argument has yet occurred).
  - Play continues until no moves are possible, an illegal *CL* is made, or the focal argument is claimed in. In the first case, Phase 2 begins, else $P$ wins.
More detail

- **Moves:**
  - **What is** (WI) — requests a label to be assigned to an argument.
  - **Claim** (CL) — assign a label to an argument.

- Players take turns to make a single move, with $P$ beginning both phases.

- **Phase 2:**
  - $P$ begins by playing **CL** on a undec labelled argument.
  - $O$ plays **WI** on a undec attacker of the **CL**.
  - This repeats until no more moves can be made. $P$ wins the game if it has made at least one move during this phase, and the labelling is legal.
Example

Phase one:
\[
P : WI(a) \\
O : CL(undec(a)) \\
P : WI(g) \\
O : CL(undec(g)) \\
P : WI(b) \\
O : CL(undec(b)) \\
P : WI(e) \\
O : CL(out(e)) \\
P : WI(f) \\
O : CL(in(f))
\]
Example

Phase two:

\[ P : \text{CL}(\text{in}(g)) \]
\[ O : \text{WI}(b) \]

\[ P : \text{CL}(\text{out}(b)) \]
\[ O : \text{WI}(a) \]

\[ P : \text{CL}(\text{in}(a)) \]
\[ O : \text{WI}(g) \]

\[ P : \text{CL}(\text{out}(g)) \]

- \( P \) contradicts itself in Phase 2, and \( O \) therefore wins — \( a \) is not skeptically preferred.
Example 2

Phase one:

- **P**: WI(d)
- **O**: CL(undec(d))
- **P**: WI(c)
- **O**: CL(undec(c))
- **P**: WI(b)
- **O**: CL(undec(b))
- **P**: WI(a)
- **O**: CL(undec(a))
Example 2

Phase two:

\( P : CL(in(d)) \)
\( O : WI(c) \)
\( P : CL(out(c)) \)
\( O : WI(b) \)
\( P : CL(in(b)) \)
\( O : WI(a) \)
\( P : CL(out(a)) \)

- In Phase 2, \( P \) successfully changes an \textit{undec} argument to \textit{in}, and therefore wins; \( d \) is skeptically preferred.
What’s going on?

- In phase 1, $O$ identifies a complete labelling where the focal argument is not in. If this is a preferred extension, then $O$ should win the game, otherwise, they’ve cheated.

- Phase 2 allows $P$ to prove that $O$’s labelling (in phase 1) is not a preferred labelling.

- Core result: there is a winning strategy for $P$ or $O$ depending on whether the argument is or isn’t skeptically preferred.

- Without perfect knowledge, this becomes a tree based discussion, requiring all possible paths to be explored.

- But in many applications, one party has perfect knowledge, reducing real world complexity.
Observations

- All proof dialogues incrementally assign a labelling to arguments.
- There is an implicit assumption that participants are cooperatively exploring the (shared) argument graph (as they know what questions are legal).
- Current work involves removing this assumption, but current results indicate that in the worst case, all arguments and attackers must be exchanged to obtain soundness and completeness, reducing to existing work.
- Since all attackers for an argument must be explored, there’s a question of cognitive load in human-centric applications over large graphs. Exploration is taking place regarding heuristics to allow short-circuiting, but this comes at the cost of completeness.
In a nutshell...

- Dialectical proof procedures are an alternative approach to identifying status of argument.
- Such proof procedures exist for many semantics.
- They implicitly encode algorithms used to perform labelings (including random choice and backtracking as necessary).
- Complexity (for a good algorithm) is equivalent to complexity of deciding whether a single argument is in the appropriate extension type.
- The main claim is that such proof procedures are more easily understood by non-experts.
Outline

1. Introduction
2. Abstract Argumentation
3. Proof Dialogues
4. Ranking-based Semantics
5. Structured Argumentation
6. Applications
The need for graduality

With Dung’s semantics, each argument is given an **absolute status** as it can only belong to one of the following cases:

- The argument belongs to every extension,
- the argument does not belong to any extensions,
- and the argument belongs to some extensions and not others.
The need for graduality

Moreover, we can notice that those semantics follow the following principles:\(^3\):

- **Killing**: The impact of an attack from an argument to another argument is drastic.
- **Existence**: One successful attack against an argument has the same effect on it as any number of successful attacks.
- **Flatness**: All the accepted arguments have the same level of acceptability.

---

\(^3\)Ranking-Based Semantics for Argumentation Frameworks. Amgoud and Ben Naim. SUM 2013
The need for graduality

In some contexts, the killing principle is not desirable as an attack does not necessarily kill its target, but just weakens it.

\( a \): The patient should have a surgery since he has cancer.

\( b \): The statistics show that the probability that a surgery will improve the state of the patient is low.

The attack from \( b \) should weaken \( a \) and not kill it. The doctor may still choose to do the surgery since it gives a chance to the patient.
The need for graduality

The same applies for the existence principle.

- **a**: John should buy this car since it is made by Peugeot and their cars are affordable and reliable.
- **b₁**: The engines of Peugeot cars break down before 300,000km.
- **b₂**: The airbags of Peugeot cars are not reliable.
- **b₃**: The spare part is very expensive.

The acceptability of **a** should decrease after receiving **b₁** to **b₃**.
The need for graduality

As a result, new semantics must be considered with the following considerations:

- **Weakening**: Arguments should not be killed and only weakened.
- **Attack sensitivity**: The more an argument is “attacked”, the greater the decrease in its acceptability.
- **Graduality**: There is an arbitrarily large number of degrees of acceptability.
A ranking-based semantics $\sigma$ is functions that transform any $AF = (\mathcal{A}, \mathcal{R})$ into a total, transitive, and binary order (ranking) $\preceq^{AF}_\sigma$ on arguments$^3$.

---

$^3$We will use $\preceq_\sigma$ when the argumentation framework is clear from the context.
A ranking-based semantics $\sigma$ is functions that transform any argumentation framework $AF = (\mathcal{A}, \mathcal{R})$ into a total, transitive, and binary order (ranking) $\preceq_{\sigma}^{AF}$ on arguments $^3$. 

Example:

$\begin{align*}
    b &\prec_{\sigma} a &\prec_{\sigma} c &\prec_{\sigma} h, d, e, g
\end{align*}$

$^3$We will use $\preceq_{\sigma}$ when the argumentation framework is clear from the context.
Categoriser ranking-based semantics

The categoriser ranking-based semantics\(^4\) is the function \(\sigma_{Cat}\) that transform any \(AF = (A, R)\) into \(\preceq_{\sigma_{Cat}}\) such that for all \(a, b \in A\), \(a \preceq_{\sigma_{Cat}} b\) iff \(Cat(a) \leq Cat(b)\), where:

\[
Cat(a) = \begin{cases} 
1 & \text{if } \not\exists (c, a) \in R \\
\frac{1}{1 + \sum_{(c, a) \in R} Cat(c)} & \text{otherwise}
\end{cases}
\]

Categoriser ranking-based semantics

The **categoriser ranking-based semantics**\(^4\) is the function \(\sigma_{\text{Cat}}\) that transform any \(AF = (A, R)\) into \(\preceq_{\sigma_{\text{Cat}}}\) such that for all \(a, b \in A\), \(a \preceq_{\sigma_{\text{Cat}}} b\) iff \(\text{Cat}(a) \leq \text{Cat}(b)\), where:

\[
\text{Cat}(a) = \begin{cases} 
1 & \text{if } \not\exists (c, a) \in R \\
\frac{1}{1 + \sum_{(c,a) \in R} \text{Cat}(c)} & \text{otherwise}
\end{cases}
\]

The discussion-based semantics is the function $\sigma_{\text{Dis}}$ that transform any $AF = (\mathcal{A}, \mathcal{R})$ into $\preceq_{\sigma_{\text{Dis}}}$ such that for all $a, b \in \mathcal{A}$, $a \preceq_{\sigma_{\text{Dis}}} b$ iff $\text{Dis}(a) \geq_{\text{lex}} \text{Dis}(b)$, where $\text{Dis}(a) = (\text{Dis}_1(a), \text{Dis}_2(a), \ldots)$ and

$$\forall i > 0, \text{Dis}_i(a) = \begin{cases} -|\{\text{path}(b, a) \text{ of length } i\}| & \text{if } i \text{ is even} \\ |\{\text{path}(b, a) \text{ of length } i\}| & \text{otherwise} \end{cases}$$

---

\(^5\) Amgoud and Ben-Naim. Ranking-based semantics for argumentation frameworks. SUM (2013)
The discussion-based semantics\textsuperscript{5} is the function $\sigma_{Dis}$ that transform any $AF = (A, R)$ into $\preceq_{\sigma_{Dis}}$ such that for all $a, b \in A$, $a \preceq_{\sigma_{Dis}} b$ iff $\text{Dis}(a) \geq_{\text{lex}} \text{Dis}(b)$, where $\text{Dis}(a) = (\text{Dis}_1(a), \text{Dis}_2(a), \ldots)$ and

$$\forall i > 0, \text{Dis}_i(a) = \begin{cases} -|\{\text{path}(b, a) \text{ of length } i\}| & \text{if } i \text{ is even} \\ |\{\text{path}(b, a) \text{ of length } i\}| & \text{otherwise} \end{cases}$$

\textsuperscript{5}Amgoud and Ben-Naim. Ranking-based semantics for argumentation frameworks. SUM (2013)
The burden-based semantics is the function $\sigma_{Bur}$ that transform any AF $=(A, R)$ into $\preceq_{\sigma_{Bur}}$ such that for all $a, b \in A$, $a \preceq_{\sigma_{Bur}} b$ iff $Bur(a) \geq_{lex} Bur(b)$, where $Bur(a) = (Bur_0(a), Bur_1(a), \ldots)$ and

$$\forall i > 0, Bur_i(a) = \begin{cases} 1 & \text{if } i = 0 \\ 1 + \sum_{(c,a) \in R} \frac{1}{Bur_{i-1}(c)} & \text{otherwise} \end{cases}$$

---

6 Amgoud and Ben-Naim. Ranking-based semantics for argumentation frameworks. SUM (2013)
The burden-based semantics\(^6\) is the function \(\sigma_{Bur}\) that transform any AF = (\(A, \mathcal{R}\)) into \(\leq_{\sigma_{Bur}}\) such that for all \(a, b \in A\), \(a \preceq_{\sigma_{Bur}} b\) iff \(Bur(a) \geq_{\text{lex}} Bur(b)\), where \(Bur(a) = (Bur_0(a), Bur_1(a), \ldots)\) and

\[
\forall i > 0, Bur_i(a) = \begin{cases} 
1 & \text{if } i = 0 \\
1 + \sum_{(c,a) \in \mathcal{R}} \frac{1}{Bur_{i-1}(c)} & \text{otherwise}
\end{cases}
\]

\[\begin{array}{|c|c|c|c|c|c|}
\hline
 & a & b & c & d & e \\
\hline
Bur_0 & 1 & 1 & 1 & 1 & 1 \\
\hline
Bur_1 & 3 & 1 & 2 & 2 & 3 \\
\hline
Bur_2 & 2.5 & 1 & 2 & 1.33 & 1.83 \\
\hline
\end{array}\]

\(a \prec_{\sigma_{Bur}} e \prec_{\sigma_{Bur}} c \prec_{\sigma_{Bur}} d \prec_{\sigma_{Bur}} b\)

---

\(^6\) Amgoud and Ben-Naim. Ranking-based semantics for argumentation frameworks. SUM (2013)
Exercise

Questions and answers are online at:

itempool.com/BrunoYun/live
Many other ranking-based semantics have been defined in the literature with different behaviour and logical properties.

Which ranking-based semantics should I use?

---

Many other ranking-based semantics have been defined in the literature with different behaviour and logical properties.

Which ranking-based semantics should I use?

Several desirable properties have been defined and used to compare them\(^7\). In the following slides, we recall some of them.

\(^7\)Bonzon et al. A Comparative Study of Ranking-Based Semantics for Abstract Argumentation. AAAI (2016)
Abstraction

The ranking on arguments obtained from $\sigma$ should be defined only on the basis of the attacks between arguments. If there is an isomorphism $\gamma$ from $AF = (\mathcal{A}, \mathcal{R})$ to $AF'$ then $\forall x, y \in \mathcal{A}, x \preceq^AF y$ iff $\gamma(x) \preceq^AF' \gamma(y)$. 

Example:

$d \prec^AF c \prec^AF b$, $d$ 

Bruno Yun (UoA, UK)
Abstraction

The ranking on arguments obtained from $\sigma$ should be defined only on the basis of the attacks between arguments. If there is an isomorphism $\gamma$ from $AF = (A, \mathcal{R})$ to $AF'$ then $\forall x, y \in A, x \preceq^\sigma y$ iff $\gamma(x) \preceq_{\sigma}^\gamma \gamma(y)$.

Example:

$$a \prec^\sigma c \prec^\sigma b, d \iff a' \prec_{\sigma}^\gamma c' \prec_{\sigma}^\gamma b', d'$$
Void precedence

A non-attacked argument should be ranked strictly higher than any attacked argument.
For any given $AF = (A, R)$, $\forall a, b \in A$ s.t. $\not\exists (c, a) \in R$ and $\exists (d, b) \in R$ then $b \prec AF a$. 
A non-attacked argument should be ranked strictly higher than any attacked argument.

For any given $AF = (A, R)$, $\forall a, b \in A$ s.t. $\neg \exists (c, a) \in R$ and $\exists (d, b) \in R$ then $b \prec_{\sigma}^{AF} a$.

**Example:**

```
d
  
  a
  
  c
```

$d, a \prec_{\sigma}^{AF} c$
Cardinality precedence

The greater the number of *direct attackers* for an argument, the weaker the level of acceptability of this argument.

For any given $AF = (A, R)$, $\forall a, b \in A$ s.t.

$$\{|(c, a) \in R | c \in A| < |{(c, b) \in R | c \in A}\}$$

then $b \prec_{AF} a$. 
The greater the number of *direct attackers* for an argument, the weaker the level of acceptability of this argument. For any given $AF = (A, R)$, $\forall a, b \in A$ s.t. $|\{(c, a) \in R \mid c \in A\}| < |\{(c, b) \in R \mid c \in A\}|$ then $b \prec^A_F a$.

**Example:**

![Diagram](https://via.placeholder.com/150)

$a \prec^A_F b \prec^A_F c$
Distributed defense precedence

The best defense is when each defender attacks a distinct attacker. For any \( AF = (A, R) \), \( \forall a, b \in A \) s.t. they have the same number of direct attackers and defenders, if the defense of \( a \) is simple\(^8\) and distributed\(^9\) whereas the defense of \( b \) is simple but not distributed then \( b \prec AF a \)

---

\(^8\)every defender attacks exactly one direct attacker
\(^9\)every direct attacker is attacked by at most one argument
Distributed defense precedence

The best defense is when each defender attacks a distinct attacker. For any $AF = (A, R)$, $\forall a, b \in A$ s.t. they have the same number of direct attackers and defenders, if the defense of $a$ is simple\(^8\) and distributed\(^9\) whereas the defense of $b$ is simple but not distributed then $b \prec^{AF}_\sigma a$

Example:

\[\begin{array}{c}
d_1 \rightarrow c_1 \\
d_2 \rightarrow c_2 \\
d_3 \rightarrow c_3 \\
d_4 \rightarrow c_4 \\
\end{array}\]

\[\begin{array}{c}
a \\
b \\
\end{array}\]

\(^8\) every defender attacks exactly one direct attacker

\(^9\) every direct attacker is attacked by at most one argument
Comparing ranking-based semantics

The three ranking-based semantics satisfy different principles as shown from the next table.

<table>
<thead>
<tr>
<th>Principle</th>
<th>Cat</th>
<th>Dbs</th>
<th>Bbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstraction</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Void precedence</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cardinality precedence</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Dist. Def. precedence</td>
<td>×</td>
<td>×</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table: Satisfaction of principles by ranking-based semantics

Other ranking-based semantics and several other principles are studied in the work of Bonzon et al. (2016).
Where are we?

- Given a set of arguments which defeat each other, we can determine which should be considered justified according to different semantics.
- Dialogical proof procedures can be used to explain why a specific argument is justified (to users).
- Ranking-based semantics can be to offer more graduality.
- But where do arguments come from?
Outline

1. Introduction
2. Abstract Argumentation
3. Proof Dialogues
4. Ranking-based Semantics
5. Structured Argumentation
6. Applications
Structured, or instantiated, argumentation systems describe where arguments come from, and which arguments and conclusions are considered justified.

Process:
- From the knowledge base, generate arguments
- Identify defeats
- Evaluate using semantics
- Take the conclusions of the justified arguments

There are myriad systems out there, we will focus on a simple version (ASPIC-) of the popular ASPIC+ framework.
Within ASPIC-, a knowledge base consists of a set of
- Strict rules of the form $a, b \rightarrow r$
- Defeasible rules of the form $a, b \Rightarrow r$

Strict rules cannot be challenged (e.g., "the speed of light is 299792.458m/s").

Defeasible rules can be (e.g., "This is the best medicine for disease X").

We will illustrate ASPIC- using a propositional language.
Given a rule $\phi \rightarrow \psi$

- $\phi$ are the rule’s premises, made up of zero or more propositions.
- $\psi$ is the rule’s conclusion, and consists of a single proposition.
- Rules with empty premises are referred to as (defeasible) facts.
Contraposition

- Strict rules are assumed to be closed under contraposition.
- If \( a \rightarrow b \), then the rule \( \neg b \rightarrow \neg a \) must be present in the knowledge base.
- And in general \( a_1, \ldots, a_n \rightarrow b \) requires
  \( \neg b, a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \rightarrow \neg a_i \).
Arguments are defined recursively.

- $A_1, \ldots, A_n \rightarrow / \Rightarrow \psi$ is an argument if
  - $A_1, \ldots, A_n$ are arguments, for $n \geq 0$
  - and there is some rule $r$ such that $\text{Conc}(A_1), \ldots, \text{Conc}(A_n) \rightarrow / \Rightarrow \psi$

- If so
  - $\text{Conc}(A) = \psi$
  - $\text{Sub}(A) = \text{Sub}(A_1) \cup \ldots \cup \text{Sub}(A_n) \cup \{A\}$
  - We say that $r$ is $A$'s **top rule**.
From rules to arguments

- $\text{DefRules}(A)$ denotes the defeasible rules used in an argument $A$.
- For a strict argument (i.e., one whose toprule is strict), the last defeasible rules is the set of last defeasible rules of its constituent arguments; for a defeasible argument, it is the argument’s top rule.
Exercise 4

Given the knowledge base with the following set of rules

<table>
<thead>
<tr>
<th>Strict rules</th>
<th>Defeasible rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_1: \rightarrow c$</td>
<td>$r_3: \Rightarrow a$</td>
</tr>
<tr>
<td>$r_2: d \rightarrow \neg b$</td>
<td>$r_4: \Rightarrow b$</td>
</tr>
<tr>
<td>$r_5: a, b \Rightarrow \neg c$</td>
<td>$r_6: c \Rightarrow d$</td>
</tr>
<tr>
<td>$r_7: d \Rightarrow \neg a$</td>
<td>$r_8: \neg b \Rightarrow \neg r_6$</td>
</tr>
</tbody>
</table>

Write down the arguments and the last defeasible rules for each one.
Exercise 4

<table>
<thead>
<tr>
<th>Strict rules</th>
<th>Defeasible rules</th>
<th>Arguments</th>
<th>Last Def. R.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_1: \rightarrow c$</td>
<td>$r_3: \Rightarrow a$</td>
<td>$A_1: \rightarrow c$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>$r_2: d \rightarrow \neg b$</td>
<td>$r_4: \Rightarrow b$</td>
<td>$A_2: \Rightarrow b$</td>
<td>${r_4}$</td>
</tr>
<tr>
<td>$r_5: a, b \Rightarrow \neg c$</td>
<td>$r_6: c \Rightarrow d$</td>
<td>$A_3: \Rightarrow a$</td>
<td>${r_3}$</td>
</tr>
<tr>
<td>$r_7: d \Rightarrow \neg a$</td>
<td>$r_8: \neg b \Rightarrow \neg r_6$</td>
<td>$A_4: A_2, A_3 \Rightarrow \neg c$</td>
<td>${r_5}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$A_5: A_1 \Rightarrow d$</td>
<td>${r_6}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$A_6: A_5 \rightarrow \neg b$</td>
<td>${r_6}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$A_7: A_5 \Rightarrow \neg a$</td>
<td>${r_7}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$A_8: A_6 \Rightarrow \neg r_6$</td>
<td>${r_8}$</td>
</tr>
</tbody>
</table>
Recall that a rule undercuts another if the application of the former invalidates the application of the latter.

This is captured by naming all rules.

An argument $A$ undercuts an argument $B$ on $B'$ iff $Conc(A) = \neg n(r)$ for some $B' \in Sub(B)$ such that $B''$'s top rule $r$ is defeasible.
Rebuttal

- A rebuts argument $B$ on $B'$ iff $\text{Conc}(A) = \neg \text{Conc}(B')$ for some $B' \in \text{Sub}(B)$.

- In other words, if a conclusion of an argument is inconsistent with another argument, then a rebut exists.
Exercise 5

What are the attacks (undercuts and/or rebuts)?

<table>
<thead>
<tr>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1$: $\rightarrow c$</td>
</tr>
<tr>
<td>$A_2$: $\Rightarrow b$</td>
</tr>
<tr>
<td>$A_3$: $\Rightarrow a$</td>
</tr>
<tr>
<td>$A_4$: $A_2, A_3 \Rightarrow \neg c$</td>
</tr>
<tr>
<td>$A_5$: $A_1 \Rightarrow d$</td>
</tr>
<tr>
<td>$A_6$: $A_5 \rightarrow \neg b$</td>
</tr>
<tr>
<td>$A_7$: $A_5 \Rightarrow \neg a$</td>
</tr>
<tr>
<td>$A_8$: $A_6 \Rightarrow \neg r_6$</td>
</tr>
</tbody>
</table>
Exercise 5

What are the attacks (undercuts and/or rebuts)?

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Rebuts</th>
<th>Undercuts</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1$: $\rightarrow c$</td>
<td>$(A_1, A_4)$</td>
<td>$(A_8, A_5)$</td>
</tr>
<tr>
<td>$A_2$: $\Rightarrow b$</td>
<td>$(A_4, A_1)$</td>
<td>$(A_8, A_8)$</td>
</tr>
<tr>
<td>$A_3$: $\Rightarrow a$</td>
<td>$(A_2, A_6)$</td>
<td>$(A_8, A_6)$</td>
</tr>
<tr>
<td>$A_4$: $A_2, A_3 \Rightarrow \neg c$</td>
<td>$(A_7, A_3)$</td>
<td>$(A_8, A_7)$</td>
</tr>
<tr>
<td>$A_5$: $A_1 \Rightarrow d$</td>
<td>$(A_7, A_4)$</td>
<td></td>
</tr>
</tbody>
</table>
From attacks to defeats

- Undercuts and rebuts define attacks between arguments.
- However, some of these attacks may not be appropriate, as some arguments may be preferred (seen as more valid) than others.
- An argument $A$ defeats $B$ if $A$ undercuts $B$, or $A$ rebuts $B$ on $B'$ and $A$ is not strictly less preferred than $B'$.
- But where do preferences come from?
Preferences

- Preferences originate from preferences over rules.
- We assume that strict rules are preferred over any defeasible rule.
- Given two sets of rules $R_1, R_2$ we can say that $R_1$ is more preferred than $R_2$
  - according to the **elitist principle** if there is some rule $a \in R_1$ such that $\forall b \in R_2, a \geq b$.
  - according to the **democratic principle** if $\forall a \in R_1$, it is the case that $\exists b \in R_2$ such that $a \geq b$.
- So we can compare sets of rules and determine which are more preferred.
Preferences

- to **lift** this to arguments, we can
  - compare all defeasible rules used in the arguments (the *weakest link* principle).
  - compare the last defeasible rules used in the arguments (the *last link* principle).
- To move from attacks to defeats, we must decide on elitist vs democratic, and weakest link vs last link.
Drawing conclusions

- We are now at the point where we have arguments and defeats, i.e., an abstract argumentation framework.
- We can evaluate this according to some semantics.
- And identify justified conclusions.
- Therefore, in order to perform reasoning we need
  - A knowledge base (made up of rules and preferences over rules).
  - A principle for aggregating rule preferences (democratic vs elitist).
  - A principle for selecting rule preferences over arguments (last link vs weakest link).
  - A semantics for evaluating arguments.
- One new concept: **universal** semantics require a conclusion to appear in all extensions, though it could originate from different arguments.
Back to rebut

\[ A : (\Rightarrow a) \rightarrow b \quad \text{and} \quad B : (\rightarrow c) \Rightarrow \neg b \]

- We defined rebut as follows: \( A \) rebuts argument \( B \) on \( B' \) iff \( \text{Conc}(A) = \neg \text{Conc}(B') \) for some \( B' \in \text{Sub}(B) \).
- Thus, \( A \) and \( B \) rebut each other. We rely on preferences (over defeasible rules) to ensure that attacks become defeats in an appropriate manner.
- An alternative approach involves \textit{restricted} rebut.
- Here, \( A \) would rebut \( B \), but not vice-versa, as such rebut requires the \textit{top} rule of the attacked argument to be defeasible.
Why Restricted Rebut?

\[ S = \{ \rightarrow wa; \rightarrow wb; \rightarrow wc; a, b \rightarrow \neg c; b, c \rightarrow \neg a; a, c \rightarrow \neg b \} \]

\[ D = \{ wa \Rightarrow a; wb \Rightarrow b; wc \Rightarrow c \} \]

\[ A_1 : (wa) \Rightarrow a \]
\[ A_2 : (wb) \Rightarrow b \]
\[ A_3 : (wc) \Rightarrow c \]
\[ A_4 : A_2, A_3 \rightarrow \neg a \]
\[ A_5 : A_1, A_3 \rightarrow \neg b \]
\[ A_6 : A_1, A_2 \rightarrow \neg c \]
Why Restricted Rebut?

\[ S = \{ \rightarrow wa; \rightarrow wb; \rightarrow wc; a, b \rightarrow \neg c; b, c \rightarrow \neg a; a, c \rightarrow \neg b \} \]

\[ D = \{ wa \Rightarrow a; wb \Rightarrow b; wc \Rightarrow c \} \]

\[ A_1 : (wa) \Rightarrow a \]
\[ A_2 : (wb) \Rightarrow b \]
\[ A_3 : (wc) \Rightarrow c \]
\[ A_4 : A_2, A_3 \rightarrow \neg a \]
\[ A_5 : A_1, A_3 \rightarrow \neg b \]
\[ A_6 : A_1, A_2 \rightarrow \neg c \]

\[ \begin{array}{ccc}
  a & b & c \\
  \neg a & \neg b & \neg c 
\end{array} \]
Why Restricted Rebut?

\[ S = \{ \rightarrow wa; \rightarrow wb; \rightarrow wc; a, b \rightarrow \neg c; b, c \rightarrow \neg a; a, c \rightarrow \neg b \} \]
\[ D = \{ wa \Rightarrow a; wb \Rightarrow b; wc \Rightarrow c \} \]

\[ A_1 : (wa) \Rightarrow a \]
\[ A_2 : (wb) \Rightarrow b \]
\[ A_3 : (wc) \Rightarrow c \]
\[ A_4 : A_2, A_3 \rightarrow \neg a \]
\[ A_5 : A_1, A_3 \rightarrow \neg b \]
\[ A_6 : A_1, A_2 \rightarrow \neg c \]
Why Restricted Rebut?

\[
S = \{ \rightarrow wa; \rightarrow wb; \rightarrow wc; a, b \rightarrow \neg c; b, c \rightarrow \neg a; a, c \rightarrow \neg b \} \\
D = \{ wa \Rightarrow a; wb \Rightarrow b; wc \Rightarrow c \}
\]

\[A_1 : (wa) \Rightarrow a\]
\[A_2 : (wb) \Rightarrow b\]
\[A_3 : (wc) \Rightarrow c\]
\[A_4 : A_2, A_3 \rightarrow \neg a\]
\[A_5 : A_1, A_3 \rightarrow \neg b\]
\[A_6 : A_1, A_2 \rightarrow \neg c\]
Why Restricted Rebut?

\[ S = \{ \rightarrow wa; \rightarrow wb; \rightarrow wc; a, b \rightarrow \neg c; b, c \rightarrow \neg a; a, c \rightarrow \neg b \} \]

\[ D = \{ wa \Rightarrow a; wb \Rightarrow b; wc \Rightarrow c \} \]

\[ A_1 : (wa) \Rightarrow a \]
\[ A_2 : (wb) \Rightarrow b \]
\[ A_3 : (wc) \Rightarrow c \]
\[ A_4 : A_2, A_3 \rightarrow \neg a \]
\[ A_5 : A_1, A_3 \rightarrow \neg b \]
\[ A_6 : A_1, A_2 \rightarrow \neg c \]
Why Unrestricted Rebut?
Why Unrestricted Rebut?

Is the problem the grounded extension?

(What about $\neg a \land \neg b$?)
Why Unrestricted Rebut?

Diagram showing a structured argumentation with nodes labeled with propositions and arrows indicating the relationships. The diagram illustrates how different propositions interact within a formal argumentation framework.
Rebutts

- Restricted Rebut ends up being too skeptical
- Unrestricted rebut doesn’t work under the preferred semantics (but does under grounded).
- This is still an open problem in argumentation.
Where are we?

- We can identify which arguments survive interaction with other arguments.
- We can identify such arguments using a dialectical proof procedure.
- We can also identify which conclusions emerge from a knowledge base.
- Lots of issues not discussed
  - uncertainty over arguments
  - argument weights
  - strategy in dialogues
  - ...

Bruno Yun (UoA, UK)
Outline

1. Introduction
2. Abstract Argumentation
3. Proof Dialogues
4. Ranking-based Semantics
5. Structured Argumentation
6. Applications
Argumentation solvers

- Working with argumentation semantics can be difficult

- Argumentation solvers are programs used reasoning with argumentation semantics on abstract argumentation graphs.

- Solvers are compared against each other at the International Competition on Computational Models of Argumentation (ICCMA)

  http://argumentationcompetition.org
## Argumentation solvers

| Dynamic      | CO DC | CO DS | CO SE | CO EE | PR DC | PR DS | PR SE | PR EE | ST DC | ST DS | ST SE | ST EE | SST DC | SST DS | SST SE | SST EE | STG DC | STG DS | STG SE | STG EE | GR DC | GR DS | GR SE | ID DC | ID DS | ID SE |
|--------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| DREDD        | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |
| CoQuiAAS v3.0| ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |
| PYGLAF       | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |
| ASPARTIX-V19 | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |
| Yonas        | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |
| Argpref      |       | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |
| EqArgSolver  | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |
| μ-toksiA     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |
| Mace4/Prover9| ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     | ✓     |

**Figure:** ICCMA 2019 solvers and their features
A non-exhaustive list of argumentation tools is:

- **SasSy** enable users to interact with the system using dialogues and understand why a certain plan was selected for execution and why other alternatives were not selected. ([https://bitbucket.org/rkulak/sassy/src/master/](https://bitbucket.org/rkulak/sassy/src/master/))

- **CISpaces** is an argumentation tool to support situational understanding in intelligence analysis ([https://cispaces.org](https://cispaces.org))

- **NAKED** is a tool to generate, visualise and export argumentation graphs from knowledge bases ([https://gite.lirmm.fr/yun/naked](https://gite.lirmm.fr/yun/naked))
SasSy: Scrutable Autonomous Systems

**Current task:** UAV is selecting a runway.

**Next task:** UAV is landing on a long runway.

**User statements:**
- assert --> CoFG
- asserted --> CoFG
- USER> why out landOnShortRunway
- SYSTEM> Long runway is required because high fuel consumption.

**User statements:**
- why high_fuel_consumption
- SYSTEM> High fuel consumption because the centre of gravity shifted.
SasSy: Scrutable Autonomous Systems

Current task: UAV is selecting a runway.
Next task: UAV is landing on a long runway.
CISpaces

https://videopress.com/v/4B91ZwI6
NAKED

Knowledge base

%--- @Facts ---
%Mevin is a cat
cat(melvin).
%Mevin belongs to Schrodinger
belongsTo(melvin,schrodinger).
%Mevin is dead
dead(melvin).
%Mevin is alive
alive(melvin).
%--- @Rules ---
%If X is a cat and X belongs to Schrodinger then X is in
an uncertain state
uncertainState(X):- belongsTo(X,schrodinger), cat(X).
%--- @Constraints ---
%X cannot be both dead and alive
\[ \neg \text{dead}(X), \neg \text{alive}(X) \].
%X cannot be both dead and in an uncertain state
\[ \neg \text{uncertainState}(X), \neg \text{dead}(X) \].
%X cannot be both alive and in an uncertain state
\[ \neg \text{uncertainState}(X), \neg \text{alive}(X) \].

Arguments & Attacks Observer Repairs Log Dot Representation

There are 5 arguments:
A0: belongsTo(melvin,schrodinger)
A1: [ A0 A2 ] -> uncertainState(melvin)
A2: cat(melvin)
A3: dead(melvin)
A4: alive(melvin)
There are 14 attacks:
([A0 A3 ], A1)

https://youtu.be/q54iNWBZ9dY
Online argumentation tools

A non-exhaustive list of online argumentation tools is:

- Visualisations of complex argumentation debate:
  - **Kialo** (pro and con arguments, votes and comments)
  - **Debategraph** (graph representation of ideas and their relationships)
  - **Rationale** and **mindmup** for creating argument maps

- **TOAST** is an online implementation of the ASPIC+ framework (https://toast.arg-tech.org)

- **GERD** is an online tool to evaluate extended argumentation frameworks (http://gerd.dbai.tuwien.ac.at/index.php)

- **Damn!** is an online argumentation tool for reasoning with defeasible rules (https://github.com/hamhec/damn)
Donald Trump is a good President.

**Pros**
- The success of Trump's policies means that he could end up being one of the best U.S. presidents ever.
- Trump's foreign policy achievements demonstrate he is a good President.
- Trump has focused on and tried his best to keep his word for what he promised his electorate.

**Cons**
- In 2019, Trump was impeached, being only the third president in history to face charges.
- Trump has used identity politics to further divide our communities.
- Trump and his associates are connected to a host of crimes being actively investigated.

Website: https://www.kialo.com
DEMO: https://www.youtube.com/watch?v=MifNyU49_JA
Debategraph

Website: https://debategraph.org
DEMO: https://www.youtube.com/watch?v=8YGoBCVXY8U
Website: https://www.rationaleonline.com
TOAST

Website: https://toast.arg-tech.org/

Applications

TOAST

Website: https://toast.arg-tech.org/

Formal Argumentation

EASSS 2021 105 / 107
There is an acceptable argument for 'accessDenied(bob)' under grounded semantics

Arguments (Hide)

A1: snores(bob)
A2: professor(bob)
A3: A2 => accessAllowed(bob)
A4: A1 => misbehaves(bob)
A5: A4 => accessDenied(bob)

Extensions (Hide)

Website: https://toast.arg-tech.org/
Applications

GERD - Genteel Extended argumentation Reasoning Device

Website: http://gerd.dbai.tuwien.ac.at

[Graphical representation of GERD]

[Extended Argumentation Framework]

Example instance

For testing purposes, you can copy/paste the following example EAF to the input area:

```
arg(a).
arg(b).
arg(c).
arg(b1).
arg(b2).
arg(c1).
arg(c2).
att(c,b).
att(c1,b1).
att(c2,b2).
att(b,a).
d(b1,c,b2).
d(b1,c2,b2).
d(b2,c1,b1).
```

Further information about the input format is available here.

--- STARTING SOLVER (clingo) ---
Answer: 1
in(c2) in(c1) in(b2) in(b1) in(c) in(b)
Answer: 2
in(c2) in(c1) in(c) in(a)
SATISFIABLE

Models : 2
Time : 0.000
Prepare : 0.000
Prepro. : 0.000
Solving : 0.000

--- SOLVER TERMINATED (exit code 0) ---
Website: https://hamhec.github.io/damn/home
DEMO: https://www.dropbox.com/s/ehah09hk0s6j95s/INRIA%20Evaluation.mp4?dl=0