Learning Argumentation Frameworks
SUM 2022

Francesca Toni

Imperial College London

18 October 2022
Outline

1. Context (Explainable AI, Argumentation)
2. Part I: Learning Abstract Argumentation Frameworks
3. Part II: Learning Assumption-Based Argumentation Frameworks
4. Future work
Map of Explainability Approaches

From “Principles and Practice of Explainable ML”; Belle&Papantonis 2021
Context: eXplainable AI (XAI)

From “Principles and Practice of Explainable ML”; Belle&Papantonis 2021
“Looking at how humans explain to each other can serve as a useful starting point for explanation in AI”
[from Explanation in AI: Insights from the social sciences. Miller; AIJ 2019]
“Looking at how humans explain to each other can serve as a useful starting point for explanation in AI”
[from *Explanation in AI: Insights from the social sciences.* Miller; AIJ 2019]

“The majority of what might look like causal attributions turn out to look like argumentative claim-backings”
Context: Argumentation

Figure: Argumentation for KR
Context: Argumentation

- Various argumentation frameworks, e.g. Abstract Argumentation (AA) and Assumption-Based Argumentation (ABA), with lots of applications

Figure: Argumentation for KR
Context: Argumentation

Figure: Argumentation for KR

- Various argumentation frameworks, e.g. Abstract Argumentation (AA) and Assumption-Based Argumentation (ABA), with lots of applications
- Can these argumentation frameworks be learnt?
Context: Argumentation

▶ Various argumentation frameworks, e.g. Abstract Argumentation (AA) and Assumption-Based Argumentation (ABA), with lots of applications

▶ Can these argumentation frameworks be learnt?

In this talk I will present two approaches to learn AA and ABA frameworks from “examples”
Argumentation: An illustration

Am I eligible to claim for UK & European Breakdown & Recovery Assistance?

You need to think about whether the insurance meets your needs and whether you can claim when you need to.

You are covered for:

- UK and European Breakdown Assistance for account holder(s) in any private car that they are travelling in
- Anyone driving a private car registered to the account holder and which is being used with his/her permission. Where the account is in joint names then up to 2 private cars can be covered
- Assistance provided at home and on the roadside with national recovery and onward travel
- No call out limit
- No excess payable

You are not covered for:

- The cost of replacement parts and associated labour to repair the vehicle
- Private cars not registered to the account holder(s) unless the account holder(s) are in the vehicle at the time of the breakdown
- Motorcycles, motorhomes, caravans, commercial vehicles (all types), vans, pick up trucks and vehicles being used for hire and reward purposes (such as taxis)
- Vehicles that do not have a valid MOT or are not serviced or maintained in line with manufacturer guidelines
- Vehicles that are more than 7 metres in length, 2.3 metres wide, 3 metres high and weigh more than 3.5 tonnes when fully loaded

Nationwide
Argumentation: An illustration

COVERED FOR: UK/EU Breakdown Assistance for account holder(s) in any private car they are travelling in
NOT COVERED FOR: private cars not registered to the account holder(s) unless in the vehicle at the time of the breakdown

Mary: account holder traveling in friend’s car; car breaks down. Is Mary covered?
Argumentation: An illustration

COVERED FOR: UK/EU Breakdown Assistance for account holder(s) in any private car they are travelling in
NOT COVERED FOR: private cars not registered to the account holder(s) unless in the vehicle at the time of the breakdown

Mary: account holder traveling in friend’s car; car breaks down. Is Mary covered?

▶ there is an argument $c(mary)$ for Mary covered (as travelling in private car)
Argumentation: An illustration

COVERED FOR: UK/EU Breakdown Assistance for account holder(s) in any private car they are travelling in
NOT COVERED FOR: private cars not registered to the account holder(s) unless in the vehicle at the time of the breakdown

Mary: account holder traveling in friend’s car; car breaks down. Is Mary covered?

▶ there is an argument \( c(mary) \) for Mary covered (as travelling in private car)

▶ there is an objection (attack) against this argument, by an argument \( nc(mary) \) for Mary not covered (as car not registered to Mary)
Argumentation: An illustration

COVERED FOR: UK/EU Breakdown Assistance for account holder(s) in any private car they are travelling in
NOT COVERED FOR: private cars not registered to the account holder(s) unless in the vehicle at the time of the breakdown

Mary: account holder traveling in friend’s car; car breaks down. Is Mary covered?

▶ there is an argument \(c(mary)\) for Mary covered (as travelling in private car)

▶ there is an objection (attack) against this argument, by an argument \(nc(mary)\) for Mary not covered (as car not registered to Mary)

▶ there is an objection (attack) against this argument, by an argument \(in(mary)\) for Mary in car at time of breakdown
Argumentation: An illustration

COVERED FOR: UK/EU Breakdown Assistance for account holder(s) in any private car they are travelling in

NOT COVERED FOR: private cars not registered to the account holder(s) unless in the vehicle at the time of the breakdown

Mary: account holder traveling in friend’s car; car breaks down. Is Mary covered?

▶ there is an argument \( c(mary) \) for Mary covered (as travelling in private car)

▶ there is an objection (attack) against this argument, by an argument \( nc(mary) \) for Mary not covered (as car not registered to Mary)

▶ there is an objection (attack) against this argument, by an argument \( in(mary) \) for Mary in car at time of breakdown

\( c(mary) \) is (dialectically) “good”/“strong” and Mary is covered
Part I: Learning Abstract Argumentation Frameworks

1. Background (AA frameworks)
2. Problem
3. Solution

Bibliography

- Dung: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. AIJ 1995
- Cocarascu, Stylianou, Cyaras, Toni: Data-Empowered Argumentation for Dialectically Explainable Predictions. ECAI 2020
Background: AA frameworks

\[ \langle \text{Args}, \text{Att} \rangle \]

where

- \text{Args} is a set (the arguments)
- \text{Att} \subseteq \text{Args} \times \text{Args} is a binary relation over \text{Args}
Background: AA frameworks

\langle \text{Args}, \text{Att} \rangle \text{ where}

- \text{Args} is a set (the \textit{arguments})
- \text{Att} \subseteq \text{Args} \times \text{Args} is a binary relation over \text{Args}

\begin{align*}
\text{c(mary)} & \leftarrow \text{nc(mary)} \leftarrow \text{in(mary)}
\end{align*}
Background: AA frameworks

\[ \langle \text{Args}, \text{Att} \rangle \] where

- \text{Args} is a set (the arguments)
- \text{Att} \subseteq \text{Args} \times \text{Args} is a binary relation over \text{Args}

\[
\begin{align*}
\text{c(mary)} & \quad \text{\leftarrow} \quad \text{nc(mary)} & \quad \text{\leftarrow} \quad \text{in(mary)}
\end{align*}
\]

Semantics for AA=“Recipes” for determining (dialectically) “good” sets of arguments (extensions)
Background: AA frameworks

\[ \langle \text{Args}, \text{Att} \rangle \]

where

- \( \text{Args} \) is a set (the *arguments*)
- \( \text{Att} \subseteq \text{Args} \times \text{Args} \) is a binary relation over \( \text{Args} \)

\[
\begin{align*}
c(\text{mary}) &\quad \leftarrow \quad nc(\text{mary}) &\quad \leftarrow \quad in(\text{mary})
\end{align*}
\]

Semantics for AA=“Recipes” for determining (dialectically) “good” sets of arguments (*extensions*)

**Grounded extension**

- Let \( G_0 \) be the set of unattacked arguments in \( \text{Args} \).
- For each \( i \in \mathbb{N} \), let \( G_{i+1} \subseteq \text{Args} \) be the set of arguments that \( G_i \) *defends* (by attacking all arguments attacking \( G_i \)).

Then \( G = \bigcup_{i \in \mathbb{N}} G_i \) is the grounded extension of \( \langle \text{Args}, \text{Att} \rangle \).
Background: AA frameworks

\(<\text{Args}, \text{Att}\rangle\) where

- \(\text{Args}\) is a set (the \textit{arguments})
- \(\text{Att} \subseteq \text{Args} \times \text{Args}\) is a binary relation over \(\text{Args}\)

\[
c(\text{mary}) \leftarrow nc(\text{mary}) \leftarrow in(\text{mary})
\]

Semantics for AA=“Recipes” for determining (dialectically) “good” sets of arguments (\textit{extensions})

Grounded extension

- Let \(G_0\) be the set of unattacked arguments in \(\text{Args}\).
- For each \(i \in \mathbb{N}\), let \(G_{i+1} \subseteq \text{Args}\) be the set of arguments that \(G_i\) \textit{defends} (by attacking all arguments attacking \(G_i\)).

Then \(G = \bigcup_{i \in \mathbb{N}} G_i\) is the grounded extension of \(\langle \text{Args}, \text{Att}\rangle\).

\(\{c(\text{mary}), in(\text{mary})\}\) is grounded, \(\{in(\text{mary})\}\) is not
Problem: An example (thanks to Guilherme Paulino-Passos)

A public park states that: *No vehicles are allowed in the park.*
Problem: An example (thanks to Guilherme Paulino-Passos)

A public park states that: *No vehicles are allowed in the park.*

The park council deliberates about the interpretation of this rule.
Problem: An example (thanks to Guilherme Paulino-Passos)

A public park states that: *No vehicles are allowed in the park.*

The park council deliberates about the interpretation of this rule. So far, it has decided the following:

1. When a young man complained that he was not allowed to use a bicycle in the park, the council decided in his favour.
2. In a similar situation, but regarding a motorized bicycle, the council rejected the complaint.
3. When an ambulance entered the park to rescue an elderly person who was feeling sick, it was considered acceptable.
4. When an unspecified vehicle entered the park to rescue a sick person, this was also considered acceptable.
Problem: An example (thanks to Guilherme Paulino-Passos)

A public park states that: *No vehicles are allowed in the park.*

The park council deliberates about the interpretation of this rule. So far, it has decided the following:

1. When a young man complained that he was not allowed to use a bicycle in the park, the council decided in his favour.
2. In a similar situation, but regarding a motorized bicycle, the council rejected the complaint.
3. When an ambulance entered the park to rescue an elderly person who was feeling sick, it was considered acceptable.
4. When an unspecified vehicle entered the park to rescue a sick person, this was also considered acceptable.

*How will the park council decide on a new case? (How can we use the past cases to reason about a new one?)*
Problem: An example (thanks to Guilherme Paulino-Passos)

- Past cases:
  1. young man’s bicycle
  2. motorized bicycle
  3. ambulance to rescue sick elderly person
  4. unspecified vehicle to rescue sick person

- New case:
  a pickup truck enters the park in order to rescue a sick person

How will the park council decide on this new case?
(How can we use the past cases to reason about a new one?)
Problem

Given

- a dataset $D$ of cases of the form $(S, o)$
  
  ($S$ features, $o \in \{+, -\}$ outcome)

  e.g. $D = \{ (\{health\_emergency, motor, ambulance\}, +),
  (\{bicycle, motor\}, -)\}$

- $D$ is consistent iff there is no $S$ such that $(S, +), (S, -) \in D$.
  Suppose $D$ is consistent.

- a default outcome $d \in \{+, -\}$

  e.g. $d = -$
Problem

Given

- a dataset $D$ of cases of the form $(S, o)$
  
  $(S \text{ features, } o \in \{+, -\} \text{ outcome})$
  
  e.g. $D = \{(\{\text{health\_emergency, motor, ambulance}\}, +),$
  $(\{\text{bicycle, motor}\}, -)\}$

  - $D$ is consistent iff there is no $S$ such that $(S, +), (S, -) \in D.$

  Suppose $D$ is consistent.

- a default outcome $d \in \{+, -\}$
  
  e.g. $d = -$ 

Determine/Explain

- the outcome of a focus case (with features) $N$
  
  e.g. $N = \{\text{health\_emergency, motor}\}$
Solution: AA-CBR (Example)

Given

- $D = \{ (\{b\}, +), (\{h\}, +), (\{b, m\}, -), (\{h, m, a\}, +) \}$
  (note: $D$ is consistent)
- default outcome: $-$
- $N = \{ h, m \}$
Solution: AA-CBR (Example)

Given

- \( D = \{(\{b\}, +), (\{h\}, +), (\{b, m\}, -), (\{h, m, a\}, +)\} \) (note: \( D \) is consistent)
- default outcome: 
- \( N = \{h, m\} \)

\( \langle \text{Args}, \text{Att} \rangle \) is:
Solution: AA-CBR (Example)

Given

- $D = \{ (\{b\}, +), (\{h\}, +), (\{b, m\}, -), (\{h, m, a\}, +) \}$
  (note: $D$ is consistent)
- default outcome: $-$
- $N = \{ h, m \}$

$\langle \text{Args}, \text{Att} \rangle$ is:
Solution: AA-CBR (Example)

Given

- $D = \{(\{b\}, +), (\{h\}, +), (\{b, m\}, -), (\{h, m, a\}, +)\}$
  (note: $D$ is consistent)
- default outcome: $-$
- $N = \{h, m\}$

The grounded extension is $G = \{(\{h, m\}, ?), (\{h\}, +)\}$.
As $(\{\}, -) \not\in G$, the AA-CBR outcome of $(\{h, m\}, ?)$ is $+$. 

\[
\begin{align*}
\langle Args, Att \rangle &\text{ is:} \\
\{\}, - &\rightarrow (\{b, m\}, -) &\rightarrow (\{h, m\}, ?) &\rightarrow (\{h, m, a\}, +) \\
(\{b\}, +) &\rightarrow (\{h\}, +) \\
\end{align*}
\]
General definition: AA Framework

Let \( \text{Args} = D \cup \{(N, ?)\} \cup \{(\{} , d\}\) .

\[ \begin{align*}
\text{for } (X, o_X), (Y, o_Y) &\in D \cup \{(\} , d\}, (X, o_X) \text{ Att } (Y, o_Y) \text{ iff } \\
1. & o_X \neq o_Y, \text{ and } \\
2. & Y \subset X, \text{ and } \\
3. & \not\exists (Z, o_X) \text{ with } Y \subset Z \subset X \\
\end{align*} \]

\[ \begin{align*}
\text{for } (Y, o_Y) &\in D, (N, ?) \text{ Att } (Y, o_Y) \text{ iff } \\
Y &\not\subset N \\
\text{ (irrelevance) } \\
\end{align*} \]

\[ \begin{align*}
e.g. \quad (\{\text{bicycle}\}, +) &\text{ attacks } (\{} , -), \\
(\{\text{health\_emergency, motor}\}, ?) &\text{ attacks } \\
(\{\text{health\_emergency, motor, ambulance}\}, +) \\
\end{align*} \]
General definition: Outcomes

We denote the opposite of an outcome \( o \in \{+, -\} \) as \( \bar{o} \), in the intuitive way:

- \( \bar{o} = - \), if \( o = + \)
- \( \bar{o} = + \), if \( o = - \)
General definition: Outcomes

We denote the opposite of an outcome $o \in \{+, -\}$ as $\bar{o}$, in the intuitive way:

- $\bar{o} = -, \text{ if } o = +$
- $\bar{o} = +, \text{ if } o = -$

We say that the outcome for the new case $N$ is:

- $d$, if $(\{\}, d)$ is in the grounded extension $G$,
- $\bar{d}$, otherwise
Definition (Nearest cases)

For a case base $CB$ and a new case $N$, a past case $(X, o_X) \in CB$ is nearest to $N$ if $X \subseteq N$, and there is no $(Y, o_Y) \in CB$ such that $Y \subseteq N$ and $X \subset Y$.

Theorem

$G$ contains all the nearest past cases to $N$.

Theorem (Unique past case)

If there is a unique nearest case $(X, o)$ to $N$, then the AA outcome of $N$ is $o$. 
Explanations in AA-CBR

- Return nearest cases
  - typical way in CBR
  - shows conflicting evidence in past cases
- Can we do better?
  - Idea: use dispute trees
Dispute trees - default outcome

GROUNDED DISPUTE TREE
Dispute trees - non-default outcome

 maximal dispute tree
Learning AA frameworks: Beyond AA-CBR

- Tabular data (discrete)
- Unstructured data (sentiment analysis)
Part II: Learning ABA Frameworks

1. Background (ABA frameworks and Logic Programming)
2. Problem
3. Solution

Bibliography

- Proietti, Toni: Learning Assumption-based Argumentation Frameworks. IJCLR 2022
Background: ABA Frameworks

An ABA framework is a tuple $\langle L, R, A, \overline{\cdot} \rangle$ where

- $\langle L, R \rangle$ is a deductive system, with $L$ a language and $R$ a set of (inference) rules of the form $s_0 \leftarrow s_1, \ldots, s_m$ ($m \geq 0$, $s_i \in L$, for $1 \leq i \leq m$);
- $A \subseteq L$ is a (non-empty) set of assumptions;
- $\overline{\cdot}$ is a total mapping from $A$ into $L$, where $\overline{a}$ is the contrary of $a$, for $a \in A$. 

Example (using schemata)

$\langle L, R \rangle = \{ p(X), q(X), r(X), a(X), b(X) \mid X \in \{1,2\} \}$;

$R = \{ p(X) \leftarrow a(X), q(X) \leftarrow b(X), r(1) \leftarrow true \}$;

$A = \{ a(X), b(X) \}$;

$a(X) = q(X), b(X) = r(X)$.
Background: ABA Frameworks

An ABA framework is a tuple $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\mathcal{A}} \rangle$ where

- $\langle \mathcal{L}, \mathcal{R} \rangle$ is a deductive system, with $\mathcal{L}$ a language and $\mathcal{R}$ a set of (inference) rules of the form $s_0 \leftarrow s_1, \ldots, s_m$ ($m \geq 0$, $s_i \in \mathcal{L}$, for $1 \leq i \leq m$);
- $\mathcal{A} \subseteq \mathcal{L}$ is a (non-empty) set of assumptions;
- $\overline{\mathcal{A}}$ is a total mapping from $\mathcal{A}$ into $\mathcal{L}$, where $\overline{a}$ is the contrary of $a$, for $a \in \mathcal{A}$.

The ABA framework is flat if no assumptions are heads of rules.
Background: ABA Frameworks

An ABA framework is a tuple \( \langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \overline{\mathcal{A}} \rangle \) where

- \( \langle \mathcal{L}, \mathcal{R} \rangle \) is a deductive system, with \( \mathcal{L} \) a language and \( \mathcal{R} \) a set of (inference) rules of the form \( s_0 \leftarrow s_1, \ldots, s_m \) \((m \geq 0, s_i \in \mathcal{L}, \text{for } 1 \leq i \leq m)\);
- \( \mathcal{A} \subseteq \mathcal{L} \) is a (non-empty) set of assumptions;
- \( \overline{\mathcal{A}} \) is a total mapping from \( \mathcal{A} \) into \( \mathcal{L} \), where \( \overline{a} \) is the contrary of \( a \), for \( a \in \mathcal{A} \).

The ABA framework is flat if no assumptions are heads of rules.

Example (using schemata)

- \( \mathcal{L} = \{ p(X), q(X), r(X), a(X), b(X) \mid X \in \{1, 2\} \} \);
- \( \mathcal{R} = \{ p(X) \leftarrow a(X), \quad q(X) \leftarrow b(X), \quad r(1) \leftarrow \text{true} \} \);
- \( \mathcal{A} = \{ a(X), b(X) \} \);
- \( \overline{a(X)} = q(X), \quad \overline{b(X)} = r(X) \).
Flat ABA frameworks $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \neg \rangle$ where $\mathcal{L}$ is a set of atoms amount to \textit{(normal) logic programs}.
Background: Logic programming

Flat ABA frameworks $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \bot \rangle$ where $\mathcal{L}$ is a set of atoms amount to (normal) logic programs.

Example (from previous slide)

- $\mathcal{L} = \{ p(X), q(X), r(X), a(X), b(X) | X \in \{1, 2\} \};$
- $\mathcal{R} = \{ p(X) \leftarrow a(X), \quad q(X) \leftarrow b(X), \quad r(1) \leftarrow true \};$
- $\mathcal{A} = \{ a(X), b(X) \};$
- $\overline{a(X)} = q(X), \quad \overline{b(X)} = r(X).$

Example (as logic program)

$$
p(X) \leftarrow \text{not } q(X)
$$
$$
q(X) \leftarrow \text{not } r(X)
$$
$$
r(1) \leftarrow
$$
Background: (flat) ABA/Logic programming semantics

- ABA:
  - *arguments* are deductions of claims using rules and supported by assumptions,
  - *attacks* are directed at the assumptions in the support of arguments;
  - Abstract Argumentation-style extension-based semantics

Example (from earlier slide)

\[ L = \{ \ldots \}; \quad R = \{ q(X) \leftarrow b(X), r(1) \leftarrow true \}; \]

\[ A = \{ \ldots \}; \quad a(X) = q(X), b(X) = r(X). \]

- Arguments:
  \[ \{ a(X) \} \vdash a(X), \quad \{ b(X) \} \vdash q(X), \quad \{ \} \vdash r(1), \quad \text{etc} \]

- \[ \{ \} \vdash r(1) \text{ attacks } \{ b(1) \} \vdash q(1), \quad \{ b(1) \} \vdash q(1) \text{ attacks } \{ a(1) \} \vdash a(1), \quad \text{etc} \]
Background: (flat) ABA/Logic programming semantics

▶ ABA:
  ▶ *arguments* are deductions of claims using rules and supported by assumptions,
  ▶ *attacks* are directed at the assumptions in the support of arguments;
  ▶ Abstract Argumentation-style extension-based semantics

Example (from earlier slide)

\[ \mathcal{L} = \{ \ldots \}; \quad \mathcal{R} = \{ q(X) \leftarrow b(X), \quad r(1) \leftarrow true \}; \]

\[ \mathcal{A} = \{ \ldots \}; \quad \overline{a(X)} = q(X), \quad \overline{b(X)} = r(X). \]

▶ *Arguments*: \( \{ a(X) \} \vdash a(X), \quad \{ b(X) \} \vdash q(X), \quad \{ \} \vdash r(1), \)
  etc
Background: (flat) ABA/Logic programming semantics

ABA:
- *arguments* are deductions of claims using rules and supported by assumptions,
- *attacks* are directed at the assumptions in the support of arguments;
- Abstract Argumentation-style extension-based semantics

Example (from earlier slide)

\[ L = \{ \ldots \}; \quad R = \{ q(X) \leftarrow b(X), \quad r(1) \leftarrow true \}; \]
\[ A = \{ \ldots \}; \quad a(X) = q(X), \quad b(X) = r(X). \]

- **Arguments**: \( \{ a(X) \} \vdash a(X), \quad \{ b(X) \} \vdash q(X), \quad \{ \} \vdash r(1), \) etc

- \( \{ \} \vdash r(1) \) attacks \( \{ b(1) \} \vdash q(1), \)
  \( \{ b(1) \} \vdash q(1) \) attacks \( \{ a(1) \} \vdash a(1), \) etc
Background: (flat) ABA/Logic programming semantics

Various notions of “acceptable” extensions (sets of arguments)
Background: (flat) ABA/Logic programming semantics

Various notions of “acceptable” extensions (sets of arguments)

One-to-one correspondence between models of logic programs and acceptable extensions in flat ABA
e.g. well-founded model $\sim$ grounded extension
Problem: An example (Dimopoulos-Kakas 1995)

▶ Given

1) **Background knowledge** (ABA framework):

\[ R = \{ \text{bird}(X) \leftarrow \text{penguin}(X), \]
\[ \text{penguin}(X) \leftarrow \text{superpenguin}(X), \]
\[ \text{bird}(a) \leftarrow, \text{bird}(b) \leftarrow, \]
\[ \text{penguin}(c) \leftarrow, \text{penguin}(d) \leftarrow, \]
\[ \text{superpenguin}(e) \leftarrow, \text{superpenguin}(f) \leftarrow \} \]
Problem: An example (Dimopoulos-Kakas 1995)

Given
1) **Background knowledge** (ABA framework):

   \[ R = \{ \text{bird}(X) \leftarrow \text{penguin}(X), \]
   \[ \text{penguin}(X) \leftarrow \text{superpenguin}(X), \]
   \[ \text{bird}(a) \leftarrow, \text{bird}(b) \leftarrow, \]
   \[ \text{penguin}(c) \leftarrow, \text{penguin}(d) \leftarrow, \]
   \[ \text{superpenguin}(e) \leftarrow, \text{superpenguin}(f) \leftarrow \} \]

2) **Positive Examples**: \{flies(a), flies(b), flies(e), flies(f)\}
Problem: An example (Dimopoulos-Kakas 1995)

Given

1) **Background knowledge** (ABA framework):

\[ R = \{ \text{bird}(X) \leftarrow \text{penguin}(X), \]
\[ \text{penguin}(X) \leftarrow \text{superpenguin}(X), \]
\[ \text{bird}(a) \leftarrow, \text{bird}(b) \leftarrow, \]
\[ \text{penguin}(c) \leftarrow, \text{penguin}(d) \leftarrow, \]
\[ \text{superpenguin}(e) \leftarrow, \text{superpenguin}(f) \leftarrow \} \]

2) **Positive Examples**: \{flies(a), flies(b), flies(e), flies(f)\}

3) **Negative Examples**: \{flies(c), flies(d)\}
Problem: An example (Dimopoulos-Kakas 1995)

▶ Given

1) **Background knowledge** (ABA framework):

\[
\mathcal{R} = \{ \begin{align*}
& \text{bird}(X) \leftarrow \text{penguin}(X), \\
& \text{penguin}(X) \leftarrow \text{superpenguin}(X), \\
& \text{bird}(a) \leftarrow, \text{bird}(b) \leftarrow, \\
& \text{penguin}(c) \leftarrow, \text{penguin}(d) \leftarrow, \\
& \text{superpenguin}(e) \leftarrow, \text{superpenguin}(f) \leftarrow \}
\]

2) **Positive Examples**: \( \{\text{flies}(a), \text{flies}(b), \text{flies}(e), \text{flies}(f)\} \)

3) **Negative Examples**: \( \{\text{flies}(c), \text{flies}(d)\} \)

▶ Determine an ABA framework “generalising” the examples
Problem: An example (Dimopoulos-Kakas 1995)

Given

1) **Background knowledge** (ABA framework):

\[ R = \{ \text{bird}(X) \leftarrow \text{penguin}(X), \]
\[ \text{penguin}(X) \leftarrow \text{superpenguin}(X), \]
\[ \text{bird}(a) \leftarrow, \text{bird}(b) \leftarrow, \]
\[ \text{penguin}(c) \leftarrow, \text{penguin}(d) \leftarrow, \]
\[ \text{superpenguin}(e) \leftarrow, \text{superpenguin}(f) \leftarrow \} \]

2) **Positive Examples**: \{ \text{flies}(a), \text{flies}(b), \text{flies}(e), \text{flies}(f) \}

3) **Negative Examples**: \{ \text{flies}(c), \text{flies}(d) \}

Determine an ABA framework “generalising” the examples

\[ R' = \{ \text{flies}(X) \leftarrow \text{bird}(X), \alpha_1(X), \]
\[ \text{c-} \alpha_1(X) \leftarrow \text{penguin}(X), \alpha_2(X), \]
\[ \text{c-} \alpha_2(X) \leftarrow \text{superpenguin}(X) \} \cup R \]

\[ A' = \{ \alpha_1(X), \alpha_2(X) \} \quad \text{with } \overline{\alpha_i(X)}' = \text{c-} \alpha_i(X) \]
Problem: Formally

▶ $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \neg \rangle \models s$ indicates that $s \in \mathcal{L}$ is the claim of an argument accepted in all or some (stable, grounded, ...) extensions of $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \neg \rangle$. 
Problem: Formally

\[ \langle L, R, A, \bar{\_} \rangle \models s \] indicates that \( s \in L \) is the claim of an argument accepted in all or some (stable, grounded, \ldots) extensions of \( \langle L, R, A, \bar{\_} \rangle \).

An example \( e \) is covered by \( \langle L, R, A, \bar{\_} \rangle \) iff \( \langle L, R, A, \bar{\_} \rangle \models e \) and is not covered by \( \langle L, R, A, \bar{\_} \rangle \) iff \( \langle L, R, A, \bar{\_} \rangle \not\models e \).
Problem: Formally

- $\langle \mathcal{L}, \mathcal{R}, A, \neg \rangle \models s$ indicates that $s \in \mathcal{L}$ is the claim of an argument accepted in all or some (stable, grounded, ...) extensions of $\langle \mathcal{L}, \mathcal{R}, A, \neg \rangle$.

- An example $e$ is \textit{covered} by $\langle \mathcal{L}, \mathcal{R}, A, \neg \rangle$ iff $\langle \mathcal{L}, \mathcal{R}, A, \neg \rangle \models e$ and is \textit{not covered} by $\langle \mathcal{L}, \mathcal{R}, A, \neg \rangle$ iff $\langle \mathcal{L}, \mathcal{R}, A, \neg \rangle \not\models e$.

- Given background knowledge $\langle \mathcal{L}, \mathcal{R}, A, \neg \rangle$, positive examples $\mathcal{E}^+$ and negative examples $\mathcal{E}^-(\mathcal{E}^+ \cap \mathcal{E}^- = \emptyset)$, the \textbf{goal of ABA learning} is to construct $\langle \mathcal{L}', \mathcal{R}', A', \neg' \rangle$ with $\mathcal{R} \subseteq \mathcal{R}'$, $A \subseteq A'$ and $\forall \alpha \in A$, $\overline{\alpha'} = \overline{\alpha}$, such that:
Problem: Formally

- \( \langle L, R, A, \neg \rangle \models s \) indicates that \( s \in L \) is the claim of an argument accepted in all or some (stable, grounded, \ldots) extensions of \( \langle L, R, A, \neg \rangle \).

- An example \( e \) is covered by \( \langle L, R, A, \neg \rangle \) iff \( \langle L, R, A, \neg \rangle \models e \) and is not covered by \( \langle L, R, A, \neg \rangle \) iff \( \langle L, R, A, \neg \rangle \not\models e \).

- Given background knowledge \( \langle L, R, A, \neg \rangle \), positive examples \( E^+ \) and negative examples \( E^- (E^+ \cap E^- = \emptyset) \), the goal of ABA learning is to construct \( \langle L', R', A', \neg' \rangle \) with \( R \subseteq R' \), \( A \subseteq A' \) and \( \forall \alpha \in A, \overline{\alpha'} = \overline{\alpha} \), such that:
  - \( (Existence) \langle L', R', A', \neg' \rangle \) admits at least one extension (under the chosen ABA semantics),
Problem: Formally

- \( \langle L, R, A, \neg \rangle \models s \) indicates that \( s \in L \) is the claim of an argument accepted in all or some (stable, grounded, ...) extensions of \( \langle L, R, A, \neg \rangle \).

- An example \( e \) is covered by \( \langle L, R, A, \neg \rangle \) iff \( \langle L, R, A, \neg \rangle \models e \) and is not covered by \( \langle L, R, A, \neg \rangle \) iff \( \langle L, R, A, \neg \rangle \not\models e \).

- Given background knowledge \( \langle L, R, A, \neg \rangle \), positive examples \( E^+ \) and negative examples \( E^- \) (\( E^+ \cap E^- = \emptyset \)), the goal of ABA learning is to construct \( \langle L', R', A', \neg' \rangle \) with \( R \subseteq R' \), \( A \subseteq A' \) and \( \forall \alpha \in A, \overline{\alpha}' = \overline{\alpha} \), such that:
  - (Existence) \( \langle L', R', A', \neg' \rangle \) admits at least one extension (under the chosen ABA semantics),
  - (Completeness) \( \forall e \in E^+, \langle L', R', A', \neg' \rangle \models e \), and
Problem: Formally

- $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \top \rangle \models s$ indicates that $s \in \mathcal{L}$ is the claim of an argument accepted in all or some (stable, grounded, ...) extensions of $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \top \rangle$.

- An example $e$ is covered by $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \top \rangle$ iff $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \top \rangle \models e$ and is not covered by $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \top \rangle$ iff $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \top \rangle \not\models e$.

- Given background knowledge $\langle \mathcal{L}, \mathcal{R}, \mathcal{A}, \top \rangle$, positive examples $\mathcal{E}^+$ and negative examples $\mathcal{E}^- (\mathcal{E}^+ \cap \mathcal{E}^- = \emptyset)$, the goal of ABA learning is to construct $\langle \mathcal{L}', \mathcal{R}', \mathcal{A}', \top' \rangle$ with $\mathcal{R} \subseteq \mathcal{R}'$, $\mathcal{A} \subseteq \mathcal{A}'$ and $\forall \alpha \in \mathcal{A}, \overline{\alpha'} = \overline{\alpha}$, such that:
  - (Existence) $\langle \mathcal{L}', \mathcal{R}', \mathcal{A}', \top' \rangle$ admits at least one extension (under the chosen ABA semantics),
  - (Completeness) $\forall e \in \mathcal{E}^+, \langle \mathcal{L}', \mathcal{R}', \mathcal{A}', \top' \rangle \models e$, and
  - (Consistency) $\forall e \in \mathcal{E}^-, \langle \mathcal{L}', \mathcal{R}', \mathcal{A}', \top' \rangle \not\models e$. 
Solution: Transformation rules for (flat) ABA frameworks

▶ Equality Removal.
Replace a rule $\rho_1$: $H \leftarrow eq_1, Eqs, B \in R$, by rule $\rho_2$: $H \leftarrow Eqs, B$. Thus, $R' = (R \setminus \{\rho_1\}) \cup \{\rho_2\}$.

▶ Folding.
Given rules $\rho_1$: $H \leftarrow Eqs_1, B_1, B_2$ and $\rho_2$: $K \leftarrow Eqs_1, Eqs_2, B_1$ in $R$, replace $\rho_1$ by $\rho_3$: $H \leftarrow Eqs_2, K, B_2$. Thus, $R' = (R \setminus \{\rho_1\}) \cup \{\rho_3\}$.

▶ Subsumption.
Delete from $R$ subsumed rules.

▶ Rote Learning.
Given atom $p(t)$, add $\rho$: $p(X) \leftarrow X = t$ to $R$. Thus, $R' = R \cup \{\rho\}$.

▶ Assumption Introduction.
Replace $\rho_1$: $H \leftarrow Eqs, B$ in $R$ by $\rho_2$: $H \leftarrow Eqs, B, \alpha(X)$ where variables in $X$ are taken from $\text{vars}(H) \cup \text{vars}(B)$ and $\alpha(X)$ is a (possibly new) assumption with contrary $\chi(X)$. Thus, $R' = (R \setminus \{\rho_1\}) \cup \{\rho_2\}$, $A' = A \cup \{\alpha(X)\}$, $\alpha(X)' = \chi(X)$, and $\beta'$ for all $\beta \in A$. 

Solution: Transformation rules for (flat) ABA frameworks

- **Equality Removal.** Replace a rule $\rho_1 : H \leftarrow eq_1, Eqs, B$ in $\mathcal{R}$, by rule $\rho_2 : H \leftarrow Eqs, B$. Thus, $\mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_2\}$. 

- **Folding.** Given rules $\rho_1 : H \leftarrow Eqs_1, B_1, B_2$ and $\rho_2 : K \leftarrow Eqs_1, Eqs_2, B_1$ in $\mathcal{R}$, replace $\rho_1$ by $\rho_3 : H \leftarrow Eqs_2, K, B_2$. Thus, $\mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_3\}$.

- **Subsumption.** Delete from $\mathcal{R}$ subsumed rules.

- **Rote Learning.** Given atom $p(t)$, add $\rho : p(X) \leftarrow X = t$ to $\mathcal{R}$. Thus, $\mathcal{R}' = \mathcal{R} \cup \{\rho\}$.

- **Assumption Introduction.** Replace $\rho_1 : H \leftarrow Eqs, B$ in $\mathcal{R}$ by $\rho_2 : H \leftarrow Eqs, B, \alpha(X)$ where variables in $X$ are taken from $\text{vars}(H) \cup \text{vars}(B)$ and $\alpha(X)$ is a (possibly new) assumption with contrary $\chi(X)$. Thus, $\mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_2\}$, $\mathcal{A}' = \mathcal{A} \cup \{\alpha(X)\}$, and $\beta' = \beta$ for all $\beta \in \mathcal{A}$.
Solution: Transformation rules for (flat) ABA frameworks

➤ *Equality Removal.* Replace a rule \( \rho_1 : H \leftarrow eq_1, Eqs, B \) in \( \mathcal{R} \), by rule \( \rho_2 : H \leftarrow Eqs, B \). Thus, \( \mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_2\} \).

➤ *Folding.* Given rules \( \rho_1 : H \leftarrow Eqs_1, B_1, B_2 \) and \( \rho_2 : K \leftarrow Eqs_1, Eqs_2, B_1 \) in \( \mathcal{R} \), replace \( \rho_1 \) by \( \rho_3 : H \leftarrow Eqs_2, K, B_2 \). Thus, \( \mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_3\} \).
Solution: Transformation rules for (flat) ABA frameworks

- **Equality Removal.** Replace a rule \( \rho_1 : H \leftarrow eq_1, Eqs, B \) in \( \mathcal{R} \), by rule \( \rho_2 : H \leftarrow Eqs, B \). Thus, \( \mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_2\} \).

- **Folding.** Given rules \( \rho_1 : H \leftarrow Eqs_1, B_1, B_2 \) and \( \rho_2 : K \leftarrow Eqs_1, Eqs_2, B_1 \) in \( \mathcal{R} \), replace \( \rho_1 \) by \( \rho_3 : H \leftarrow Eqs_2, K, B_2 \). Thus, \( \mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_3\} \).

- **Subsumption.** Delete from \( \mathcal{R} \) subsumed rules.
Solution: Transformation rules for (flat) ABA frameworks

- **Equality Removal.** Replace a rule $\rho_1 : H \leftarrow eq_1, Eqs, B$ in $\mathcal{R}$, by rule $\rho_2 : H \leftarrow Eqs, B$. Thus, $\mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_2\}$.

- **Folding.** Given rules $\rho_1 : H \leftarrow Eqs_1, B_1, B_2$ and $\rho_2 : K \leftarrow Eqs_1, Eqs_2, B_1$ in $\mathcal{R}$, replace $\rho_1$ by $\rho_3 : H \leftarrow Eqs_2, K, B_2$. Thus, $\mathcal{R}' = (\mathcal{R} \setminus \{\rho_1\}) \cup \{\rho_3\}$.

- **Subsumption.** Delete from $\mathcal{R}$ subsumed rules.

- **Rote Learning.** Given atom $p(t)$, add $\rho : p(X) \leftarrow X = t$ to $\mathcal{R}$. Thus, $\mathcal{R}' = \mathcal{R} \cup \{\rho\}$.
Solution: Transformation rules for (flat) ABA frameworks

▶ **Equality Removal.** Replace a rule \( \rho_1 : H \leftarrow eq_1, Eqs, B \) in \( \mathcal{R} \), by rule \( \rho_2 : H \leftarrow Eqs, B \). Thus, \( \mathcal{R}' = (\mathcal{R} \setminus \{ \rho_1 \}) \cup \{ \rho_2 \} \).

▶ **Folding.** Given rules \( \rho_1 : H \leftarrow Eqs_1, B_1, B_2 \) and \( \rho_2 : K \leftarrow Eqs_1, Eqs_2, B_1 \) in \( \mathcal{R} \), replace \( \rho_1 \) by \( \rho_3 : H \leftarrow Eqs_2, K, B_2 \). Thus, \( \mathcal{R}' = (\mathcal{R} \setminus \{ \rho_1 \}) \cup \{ \rho_3 \} \).

▶ **Subsumption.** Delete from \( \mathcal{R} \) subsumed rules.

▶ **Rote Learning.** Given atom \( p(t) \), add \( \rho : p(X) \leftarrow X = t \) to \( \mathcal{R} \). Thus, \( \mathcal{R}' = \mathcal{R} \cup \{ \rho \} \).

▶ **Assumption Introduction.** Replace \( \rho_1 : H \leftarrow Eqs, B \) in \( \mathcal{R} \) by \( \rho_2 : H \leftarrow Eqs, B, \alpha(X) \) where variables in \( X \) are taken from \( \text{vars}(H) \cup \text{vars}(B) \) and \( \alpha(X) \) is a (possibly new) assumption with contrary \( \chi(X) \). Thus,

\[
\begin{align*}
\mathcal{R}' &= (\mathcal{R} \setminus \{ \rho_1 \}) \cup \{ \rho_2 \}, \\
\mathcal{A}' &= \mathcal{A} \cup \{ \alpha(X) \}, \\
\overline{\alpha(X)}' &= \chi(X), \text{ and } \overline{\beta}' = \overline{\beta} \text{ for all } \beta \in \mathcal{A}.
\end{align*}
\]
Solution: Requirements

Let $\langle L', R', A', \neg' \rangle$ be obtained by applying any of Folding, Equality Removal and Subsumption to $\langle L, R, A, \neg \rangle$ to modify rules with $p$ in the head.

If $\langle L, R, A, \neg \rangle \models p(t)$ then $\langle L', R', A', \neg' \rangle \models p(t)$. 
Solution: Requirements

- Let $\langle L', R', A', \overline{\_}' \rangle$ be obtained by applying any of Folding, Equality Removal and Subsumption to $\langle L, R, A, \overline{\_} \rangle$ to modify rules with $p$ in the head. If $\langle L, R, A, \overline{\_} \rangle \models p(t)$ then $\langle L', R', A', \overline{\_}' \rangle \models p(t)$.

- Let $p(t_1), p(t_2)$ be atoms such that $p(t_1) \neq p(t_2)$ and $\langle L, R, A, \overline{\_} \rangle \models p(t_1)$ and $\langle L, R, A, \overline{\_} \rangle \models p(t_2)$. 

Solution: Requirements

Let $\langle L', R', A', \neg' \rangle$ be obtained by applying any of Folding, Equality Removal and Subsumption to $\langle L, R, A, \neg \rangle$ to modify rules with $p$ in the head.

If $\langle L, R, A, \neg \rangle \models p(t)$ then $\langle L', R', A', \neg' \rangle \models p(t)$.

Let $p(t_1), p(t_2)$ be atoms such that $p(t_1) \neq p(t_2)$ and $\langle L, R, A, \neg \rangle \models p(t_1)$ and $\langle L, R, A, \neg \rangle \models p(t_2)$.

There exists $\langle L', R', A', \neg' \rangle$ obtained from $\langle L, R, A, \neg \rangle$ by applying Assumption Introduction to modify rules with $p$ in the head and then Rote Learning to add rules for the contraries of the assumptions, such that $\langle L', R', A', \neg' \rangle \models p(t_1)$ and $\langle L', R', A', \neg' \rangle \not\models p(t_2)$. 
An illustration

Given $\mathcal{R} = \{\text{step}(1, 2) \leftarrow, \text{step}(1, 3) \leftarrow, \text{step}(2, 4) \leftarrow, \text{step}(2, 5) \leftarrow, \text{step}(4, 6) \leftarrow, \text{step}(5, 2) \leftarrow, \text{busy}(3) \leftarrow, \text{busy}(6) \leftarrow\}$;

$\mathcal{E}^+ = \{\text{free}(1), \text{free}(2), \text{free}(5)\},$

$\mathcal{E}^- = \{\text{free}(3), \text{free}(4), \text{free}(6)\};$
An illustration

Given \( \mathcal{R} = \{ \text{step}(1, 2) \leftarrow, \text{step}(1, 3) \leftarrow, \text{step}(2, 4) \leftarrow, \text{step}(2, 5) \leftarrow, \text{step}(4, 6) \leftarrow, \text{step}(5, 2) \leftarrow, \text{busy}(3) \leftarrow, \text{busy}(6) \leftarrow \}; \)
\[ \mathcal{E}^+ = \{ \text{free}(1), \text{free}(2), \text{free}(5) \}, \]
\[ \mathcal{E}^- = \{ \text{free}(3), \text{free}(4), \text{free}(6) \}; \]

- **Rote Learning** introduces
  \[ \text{free}(X) \leftarrow X = 1 \] (1)
An illustration

Given $\mathcal{R} = \{step(1, 2) \leftarrow, step(1, 3) \leftarrow, step(2, 4) \leftarrow, step(2, 5) \leftarrow,$

$step(4, 6) \leftarrow, step(5, 2) \leftarrow, busy(3) \leftarrow, busy(6) \leftarrow\};$

$\mathcal{E}^+ = \{free(1), free(2), free(5)\},$

$\mathcal{E}^- = \{free(3), free(4), free(6)\};$

- **Rote Learning** introduces
  
  $free(X) \leftarrow X = 1 \quad (1)$

- **Folding** with the (normalised) $step(X, Y) \leftarrow X = 1, Y = 2$ in $\mathcal{R}$ gives
  
  $free(X) \leftarrow Y = 2, step(X, Y) \quad (2)$
An illustration

Given \( \mathcal{R} = \{\text{step}(1, 2) \leftarrow, \text{step}(1, 3) \leftarrow, \text{step}(2, 4) \leftarrow, \text{step}(2, 5) \leftarrow, \text{step}(4, 6) \leftarrow, \text{step}(5, 2) \leftarrow, \text{busy}(3) \leftarrow, \text{busy}(6) \leftarrow\}; \)
\[ \mathcal{E}^+ = \{\text{free}(1), \text{free}(2), \text{free}(5)\}, \]
\[ \mathcal{E}^- = \{\text{free}(3), \text{free}(4), \text{free}(6)\}; \]

- **Rote Learning** introduces
  \[ \text{free}(X) \leftarrow X = 1 \]  (1)

- **Folding** with the (normalised) \( \text{step}(X, Y) \leftarrow X = 1, Y = 2 \) in \( \mathcal{R} \) gives
  \[ \text{free}(X) \leftarrow Y = 2, \text{step}(X, Y) \]  (2)

- By **Equality Removal**, we get
  \[ \text{free}(X) \leftarrow \text{step}(X, Y) \]  (3)
An illustration (Continued)

Given $\mathcal{R} = \{\text{step}(1, 2) \leftarrow, \text{step}(1, 3) \leftarrow, \text{step}(2, 4) \leftarrow, \text{step}(2, 5) \leftarrow, \text{step}(4, 6) \leftarrow, \text{step}(5, 2) \leftarrow, \text{busy}(3) \leftarrow, \text{busy}(6) \leftarrow\}$;

$\mathcal{E}^+ = \{\text{free}(1), \text{free}(2), \text{free}(5)\}$,

$\mathcal{E}^- = \{\text{free}(3), \text{free}(4), \text{free}(6)\}$;

- Rule (3) $\text{free}(X) \leftarrow \text{step}(X, Y)$ covers $\mathcal{E}^+$ as well as $\text{free}(4) \in \mathcal{E}^-$. 

Assumption Introduction gives $\alpha(X, Y)$ with contrary $\alpha(X, Y)$ and replaces (3) by $\text{free}(X) \leftarrow \text{step}(X, Y)$.

Then, we add positive and negative examples for $\alpha(X, Y)$:

$\mathcal{E}^+ = 1 = \{c-\alpha(4, 6)\}$,

$\mathcal{E}^- = 1 = \{\text{c-}\alpha(1, 2), \text{c-}\alpha(2, 4), \text{c-}\alpha(2, 5), \text{c-}\alpha(5, 2)\}$.

$\ldots$

$\text{c-}\alpha(X, Y) \leftarrow \text{busy}(Y) (6)$

The final learnt set of rules is $\mathcal{R} \cup \{(4), (6)\}$.
An illustration (Continued)

Given \( \mathcal{R} = \{ \text{step}(1, 2) \leftarrow, \text{step}(1, 3) \leftarrow, \text{step}(2, 4) \leftarrow, \text{step}(2, 5) \leftarrow, \text{step}(4, 6) \leftarrow, \text{step}(5, 2) \leftarrow, \text{busy}(3) \leftarrow, \text{busy}(6) \leftarrow \}; \)

\[ \mathcal{E}^+ = \{ \text{free}(1), \text{free}(2), \text{free}(5) \}, \]
\[ \mathcal{E}^- = \{ \text{free}(3), \text{free}(4), \text{free}(6) \}; \]

Rule (3) \( \text{free}(X) \leftarrow \text{step}(X, Y) \) covers \( \mathcal{E}^+ \) as well as \( \text{free}(4) \in \mathcal{E}^- \). Assumption Introduction gives \( \alpha(X, Y) \) with contrary \( c-\alpha(X, Y) \) and replaces (3) by

\[ \text{free}(X) \leftarrow \text{step}(X, Y), \alpha(X, Y) \]  \hspace{1cm} (4)
An illustration (Continued)

Given $\mathcal{R} = \{ \text{step}(1, 2) \leftarrow, \text{step}(1, 3) \leftarrow, \text{step}(2, 4) \leftarrow, \text{step}(2, 5) \leftarrow, \\
\text{step}(4, 6) \leftarrow, \text{step}(5, 2) \leftarrow, \text{busy}(3) \leftarrow, \text{busy}(6) \leftarrow \}$;

$\mathcal{E}^+ = \{ \text{free}(1), \text{free}(2), \text{free}(5) \}$,

$\mathcal{E}^- = \{ \text{free}(3), \text{free}(4), \text{free}(6) \}$;

Rule (3) $\text{free}(X) \leftarrow \text{step}(X, Y)$ covers $\mathcal{E}^+$ as well as $\text{free}(4) \in \mathcal{E}^-$. Assumption Introduction gives $\alpha(X, Y)$ with contrary $c-\alpha(X, Y)$ and replaces (3) by

$\text{free}(X) \leftarrow \text{step}(X, Y), \alpha(X, Y)$

(4)

Then, we add positive and negative examples for $c-\alpha(X, Y)$:

$\mathcal{E}^+_1 = \{ c-\alpha(4, 6) \}$,

$\mathcal{E}^-_1 = \{ c-\alpha(1, 2), c-\alpha(2, 4), c-\alpha(2, 5), c-\alpha(5, 2) \}$. 
An illustration (Continued)

Given $\mathcal{R} = \{\text{step}(1, 2) \leftarrow, \text{step}(1, 3) \leftarrow, \text{step}(2, 4) \leftarrow, \text{step}(2, 5) \leftarrow, \text{step}(4, 6) \leftarrow, \text{step}(5, 2) \leftarrow, \text{busy}(3) \leftarrow, \text{busy}(6) \leftarrow\}$;

$\mathcal{E}^+ = \{\text{free}(1), \text{free}(2), \text{free}(5)\}$,

$\mathcal{E}^- = \{\text{free}(3), \text{free}(4), \text{free}(6)\}$;

- Rule (3) $\text{free}(X) \leftarrow \text{step}(X, Y)$ covers $\mathcal{E}^+$ as well as $\text{free}(4) \in \mathcal{E}^-$. Assumption Introduction gives $\alpha(X, Y)$ with contrary $c-\alpha(X, Y)$ and replaces (3) by

  $\text{free}(X) \leftarrow \text{step}(X, Y), \alpha(X, Y)$ (4)

- Then, we add positive and negative examples for $c-\alpha(X, Y)$:

  $\mathcal{E}_{1}^+ = \{c-\alpha(4, 6)\}$,

  $\mathcal{E}_{1}^- = \{c-\alpha(1, 2), c-\alpha(2, 4), c-\alpha(2, 5), c-\alpha(5, 2)\}$.

- ... $c-\alpha(X, Y) \leftarrow \text{busy}(Y)$ (6)
An illustration (Continued)

Given \( \mathcal{R} = \{ \text{step}(1, 2) \leftarrow, \text{step}(1, 3) \leftarrow, \text{step}(2, 4) \leftarrow, \text{step}(2, 5) \leftarrow, \text{step}(4, 6) \leftarrow, \text{step}(5, 2) \leftarrow, \text{busy}(3) \leftarrow, \text{busy}(6) \leftarrow \}; \)

\( \mathcal{E}^+ = \{ \text{free}(1), \text{free}(2), \text{free}(5) \}, \)

\( \mathcal{E}^- = \{ \text{free}(3), \text{free}(4), \text{free}(6) \}; \)

\( \triangleright \) Rule (3) \( \text{free}(X) \leftarrow \text{step}(X, Y) \) covers \( \mathcal{E}^+ \) as well as \( \text{free}(4) \in \mathcal{E}^- \). Assumption Introduction gives \( \alpha(X, Y) \) with contrary \( c-\alpha(X, Y) \) and replaces (3) by

\[ \text{free}(X) \leftarrow \text{step}(X, Y), \alpha(X, Y) \]  \hspace{1cm} (4)

\( \triangleright \) Then, we add positive and negative examples for \( c-\alpha(X, Y) \):

\( \mathcal{E}^+_1 = \{ c-\alpha(4, 6) \}, \)

\( \mathcal{E}^-_1 = \{ c-\alpha(1, 2), c-\alpha(2, 4), c-\alpha(2, 5), c-\alpha(5, 2) \}. \)

\( \triangleright \) …

\( c-\alpha(X, Y) \leftarrow \text{busy}(Y) \)  \hspace{1cm} (6)

The final learnt set of rules is \( \mathcal{R} \cup \{(4), (6)\}. \)
Explanations: Dispute trees?

Mary \((m)\): account holder traveling in friend’s car \((c)\); car breaks down

\[\begin{align*}
\mathcal{R} : & \quad cov(m, c) \leftarrow ah(m), tr(m, c), pr(c), not \neg cov(m, c) \\
& \quad \neg cov(m, c) \leftarrow \neg reg(c, m), not \ cov'(m, c) \\
& \quad cov'(m, c) \leftarrow in(m, c) \\
& \quad ah(m) \leftarrow \quad tr(m, c) \leftarrow \quad pr(c) \leftarrow \\
& \quad \neg reg(c, m) \leftarrow \quad in(m, c) \leftarrow \\
\mathcal{L} : & \quad \text{Herbrand base of } \mathcal{R} \text{ plus (all) NAF literals} \\
\mathcal{A} : & \quad \text{(all) NAF literals} \\
\end{align*}\]

\(\overline{not \; x} = x\) for all \(x\) in the Herbrand base of \(\mathcal{R}\)
Explanations: Dispute trees?

Mary (m): account holder traveling in friend’s car (c); car breaks down

\[ \mathcal{R} : \begin{align*}
&\text{cov}(m, c) \leftarrow ah(m), tr(m, c), pr(c), not \neg\text{cov}(m, c) \\
&\neg\text{cov}(m, c) \leftarrow \neg\text{reg}(c, m), not \text{cov}'(m, c) \\
&\text{cov}'(m, c) \leftarrow in(m, c) \\
&ah(m) \leftarrow tr(m, c) \leftarrow pr(c) \leftarrow \\
&\neg\text{reg}(c, m) \leftarrow in(m, c) \leftarrow
\end{align*} \]

\[ \mathcal{O} : \begin{align*}
&\neg\text{cov}'(m, c) \leftarrow \\
&P : \{ \neg\text{cov}(m, c) \} \vdash \text{cov}(m, c) \\
&\neg\text{cov}(m, c) \leftarrow \\
&O : \{ \text{not cov}'(m, c) \} \vdash \neg\text{cov}(m, c) \\
&\text{cov}'(m, c) \leftarrow \\
&P : \{ \} \vdash \text{cov}'(m, c)
\end{align*} \]

\[ \mathcal{L} : \text{Herbrand base of } \mathcal{R} \text{ plus (all) NAF literals} \]

\[ \mathcal{A} : \text{(all) NAF literals} \]

\[ not \overline{x} = x \text{ for all } x \text{ in the Herbrand base of } \mathcal{R} \]
Future Work

- Learning AA frameworks: non-discrete data?
Future Work

- Learning AA frameworks: non-discrete data?
- Learning ABA frameworks:
  - Formal guarantees
  - Implementation and Experiments
  - Comparison with other methods/systems to learn logic programs/argumentation frameworks
  - Learning other ABA instances (beyond logic programming)
Future Work

- Learning AA frameworks: non-discrete data?
- Learning ABA frameworks:
  - Formal guarantees
  - Implementation and Experiments
  - Comparison with other methods/systems to learn logic programs/argumentation frameworks
  - Learning other ABA instances (beyond logic programming)
- Does ABA learning generalise AA learning?
Future Work

- Learning AA frameworks: non-discrete data?
- Learning ABA frameworks:
  - Formal guarantees
  - Implementation and Experiments
  - Comparison with other methods/systems to learn logic programs/argumentation frameworks
  - Learning other ABA instances (beyond logic programming)
- Does ABA learning generalise AA learning?
- Integration with sub-symbolic machine learning
Future Work

- Learning AA frameworks: non-discrete data?
- Learning ABA frameworks:
  - Formal guarantees
  - Implementation and Experiments
  - Comparison with other methods/systems to learn logic programs/argumentation frameworks
  - Learning other ABA instances (beyond logic programming)
- Does ABA learning generalise AA learning?
- Integration with sub-symbolic machine learning
- Explanation extraction and user evaluation
- Applications