議論過程の可視化システム

高橋 和子

関西学院大学

2018 年 10 月 2 日

議論過程の可視化システム

1. PROLEG ブロック図による表示と編集
2. PROLEG から双極議論フレームワーク (BAF) への変換
3. 予測 AF にもとづく対話モデルの実行過程の表示

議論過程の可視化システム

1. PROLEG ブロック図による表示と編集
2. PROLEG から双極議論フレームワーク (BAF) への変換
3. 予測 AF にもとづく対話モデルの実行過程の表示

概要

1. グラフ上の編集機能の向上
2. 表示機能の改良と拡張

PROLEG ブロック図による表示と編集

- 佐藤らによって開発されたブロック図を拡張
- ブロック図は PROLEG プログラムにゴールを与えて実行し、実行過程を各ノードが論証，エッジが攻撃または支持とったグラフ表現 (木構造)
- ルートの論証が成り立つかどうかを計算するとともに，理由 (論証間の因果関係) を明確にする
- 編集によって各要件事実が成立するかどうかを変更したり，新たな論証が加えたときに再計算を行う

PROLEG プログラム

- `H :- B1, ..., Bn.` % 規則
  `excetion(B,E).` % 例外
  ただし，`B1,...,Bn,B,E,H` はアトム.
- 意味：B1, ..., Bn が成り立ち、かつ、E が成り立たないとき、H が成り立つ.
- `H :- B1, ..., Bn.` % 事実
- 意味：H が成り立つ.

グラフ上の編集機能の向上

- update ボタンを押したときの (編集を反映させた) ブロック図の表示　および PROLEG コードへの変換、
- また、直接 PROLEG を走らせて出力する命令を用意
- 事実の書き換えの処理。変数を定数に置き換えた場合は、OR で分岐する手前までは、全部その定数で置き換える。定数を置き換えた場合には、グラフ全体をそれに置き換える。
- undo/redo 機能の追加およびそのキーバインディング

表示機能の改良と拡張

- 構造体データの表示の切り替え機能
- メタコールに関する表示の切り替え機能
- or 関係にあるノードの表示をコピーを作らず 1 つにまとめて表示する機能

ブロック図表示の今後の拡張仕様

- 事実ベースの入力編集インタフェースの作成（可能な述語のメニューの表示と指定された述語の引数を示すテンプレートの表示）
- 可能な反論の表示と入力編集インタフェースの作成
- or 関係のノードがある場合のユーザによる指定と表示の切り替え
- モジュール化表示

目的

- 論証同士の関係を明確にする
- 推論過程の説明を明確にする

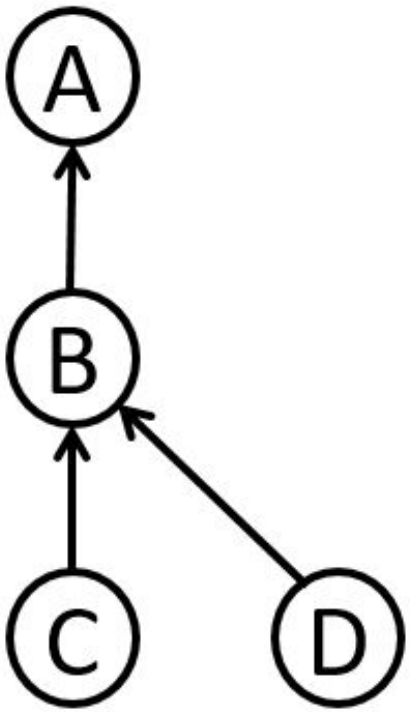
(抽象) 議論フレームワーク (Argumentation Framework, AF)

AF は論証の集合とその上の二項関係 (攻撃) で定義される

$af = \langle AR, ATT \rangle$

AR is a set, $ATT \subseteq AR \times AR$

例：$af = \langle \{A, B, C, D\}, \{(B, A), (C, B), (D, B)\} \rangle$



双極議論フレームワーク (Bipolar Argumentation Framework, BAF)

BAF は論証の集合、その上の二つの二項関係 (攻撃、支持) で定義される

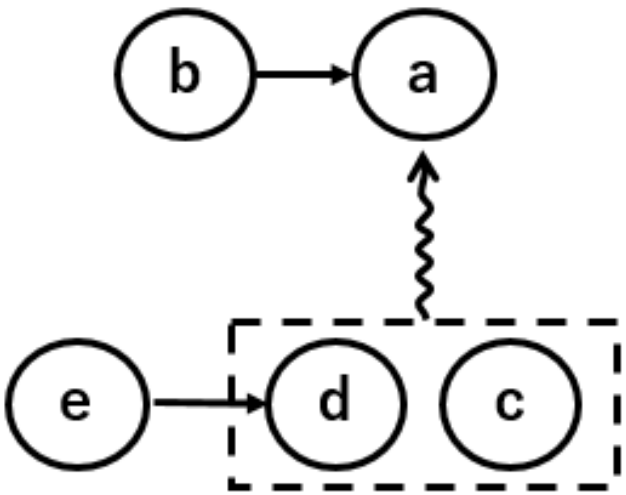
$baf = \langle AR, ATT, SUP \rangle$

$ATT \subseteq AR \times AR$, $SUP \subseteq AR \times AR$

ここでは、支持を論証の集合と論証との二項関係に拡張する。

$ATT \subseteq AR \times AR$ and $SUP \subseteq (2^{AR} \backslash \emptyset) \times AR$

例: $baf = \langle \{a, b, c, d, e\}\{(b, a), (e, d)\}\{(\{c, d\}, a)\} \rangle$.



PROLEG の意味論

- 解集合による完全モデルで定義される
- PROEG program P に対する解集合 M の定義：M が集合 $P^M = \{H \Leftarrow B_1, \ldots, B_n \in \mathcal{R} | \forall E \in \mathcal{E},$ if $head(E) =$ H then $body(E) \not\subseteq M\}$ の最小モデルである．[Satoh11]

ラベリング

- $baf = \langle AR, ATT, SUP \rangle$ に対して，ラベリング $\mathcal{L}$ とは AR から $\{in, out\}$ への関数である．
  - $\mathbf{A} \in (2^{AR} \setminus \emptyset)$ に対して，
    $\mathcal{L}(\mathbf{A}) = in$，if $\forall A \in \mathbf{A}, \mathcal{L}(A) = in$
  - $\mathcal{L}(\mathbf{A}) = out$, otherwise

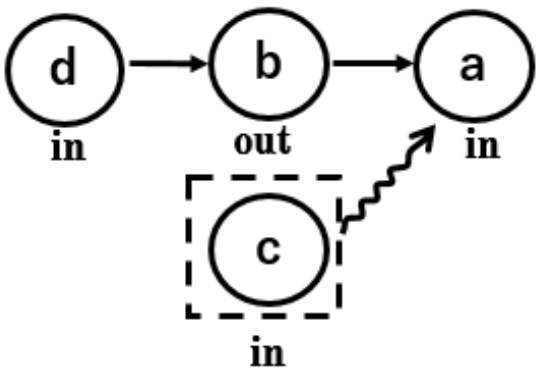
BAF の受理集合：$\{A | \mathcal{L}(A) = in\}$
まず、非循環なプログラムを対象とする

完全ラベリング

$baf = \langle AR, ATT, SUP \rangle$ において，任意の $A \in AR$ に対するラベリング $\mathcal{L}$ が以下を満たす

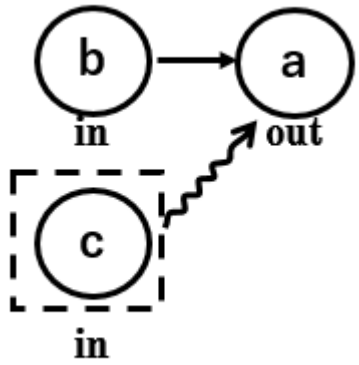
- $\forall B \in AR(\neg \mathrm{att}(B, A) \wedge) \neg \forall \mathbf{A} \subseteq AR(\mathrm{sup}(\mathbf{A}, A))$ ならば $\mathcal{L}(A) = in$.
- $(\forall B \in AR, \mathrm{att}(B, A) \Rightarrow \mathcal{L}(B) = out) \wedge (\exists \mathbf{A} \subseteq AR, \mathrm{sup}(\mathbf{A}, A) \wedge \mathcal{L}(\mathbf{A}) = in)$ ならば，$\mathcal{L}(A) = in$.
- そうでなければ $\mathcal{L}(A) = out$.

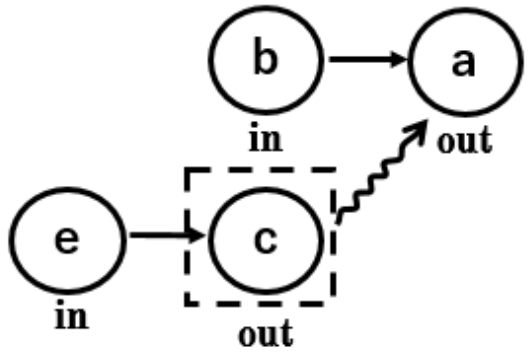
BAF の受理集合: $\{A \mid \mathcal{L}(A) = in\}$
例: $\langle \{a, b, c, d\}, \{(b, a), (d, a)\}, \{(\{c\}, a)\} \rangle$

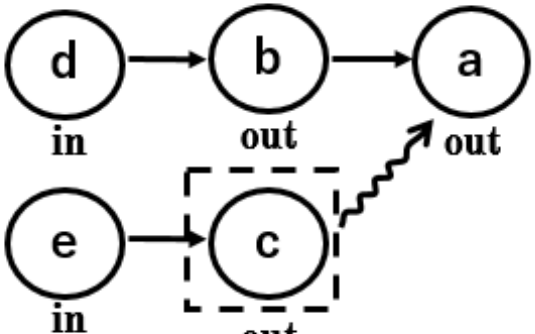


完全ラベリングの例

- $\forall B \in AR(\neg \mathrm{att}(B, A) \wedge) \neg \forall \mathbf{A} \subseteq AR(\mathrm{sup}(\mathbf{A}, A))$ ならば $\mathcal{L}(A) = in$.
- $(\forall B \in AR, \mathrm{att}(B, A) \Rightarrow \mathcal{L}(B) = out) \wedge (\exists \mathbf{A} \subseteq AR, \mathrm{sup}(\mathbf{A}, A) \wedge \mathcal{L}(\mathbf{A}) = in)$ ならば，$\mathcal{L}(A) = in$.
- そうでなければ $\mathcal{L}(A) = out$.







変換規則

PROLEG program $\langle \mathcal{R}, \mathcal{E} \rangle$ to BAF (AR, ATT, SUP)

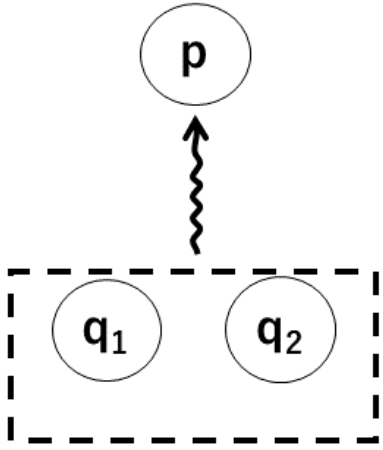
- $Atom = \bigcup_{R \in \mathcal{R}} (head(R) \cup body(R)) \cup \bigcup_{E \in \mathcal{E}} (head(E) \cup body(E))$
- $Rule = \{(body(R), head(R)) \mid R \in \mathcal{R} \wedge body(R) \neq \emptyset\}$
- $Exc = \{(B, H) \mid \mathrm{exception}(H, B) \in \mathcal{E}\}$
- $Existence = \{H \mid H \Leftarrow\in \mathcal{R}\}$
- $ExistenceSupport = \{(\{\mathrm{ex}(H)\}, H) \mid H \in Existence\}$
- $Absence = Atom \backslash (\{head(R) \mid R \in \mathcal{R}\} \cup \{head(E) \mid E \in \mathcal{E}\})$
- $AbsenceAttack = \{(\mathrm{ab}(B), B) \mid B \in Absence\}$
- $AR = Atom \cup \{\mathrm{ex}(H) \mid H \in Existence\} \cup \{\mathrm{ab}(B) \mid B \in Absence\}$
- $ATT = Exc \cup AbsenceAttack$
- $SUP = Rule \cup ExistenceSupport$

変換例:PROLEG program

```
p <= q1,q2.    % 規則
exception(q1,r).  % 例外
q2<=.    % 事実
r<=.     % 事実
```

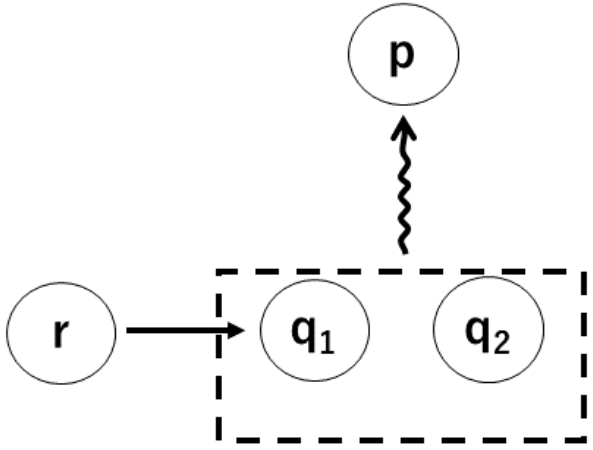
変換規則 (規則)

```
p <= q1,q2.    % 規則
```



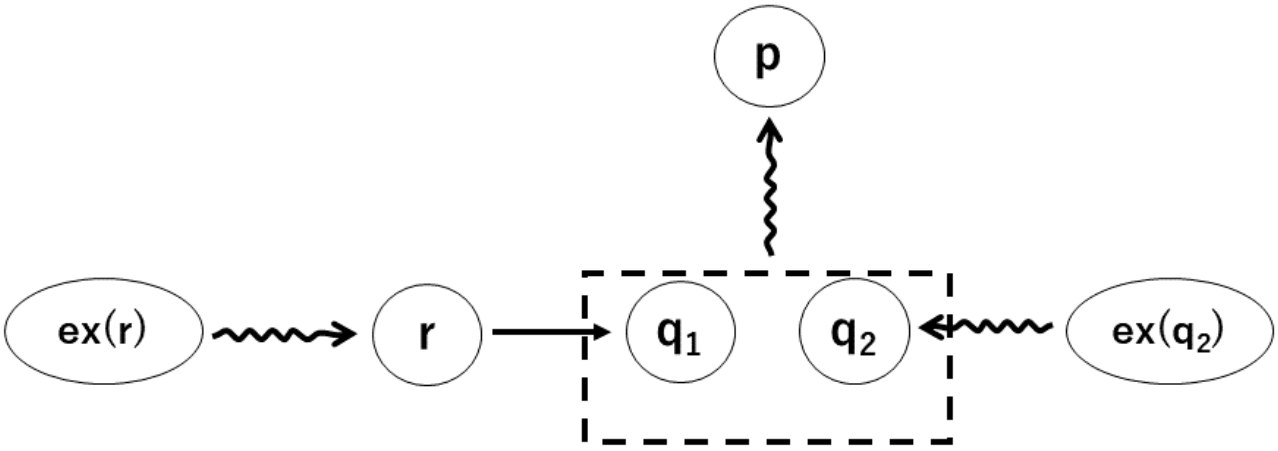
変換規則 (例外)

```
exception(q1,r).  % 例外
```



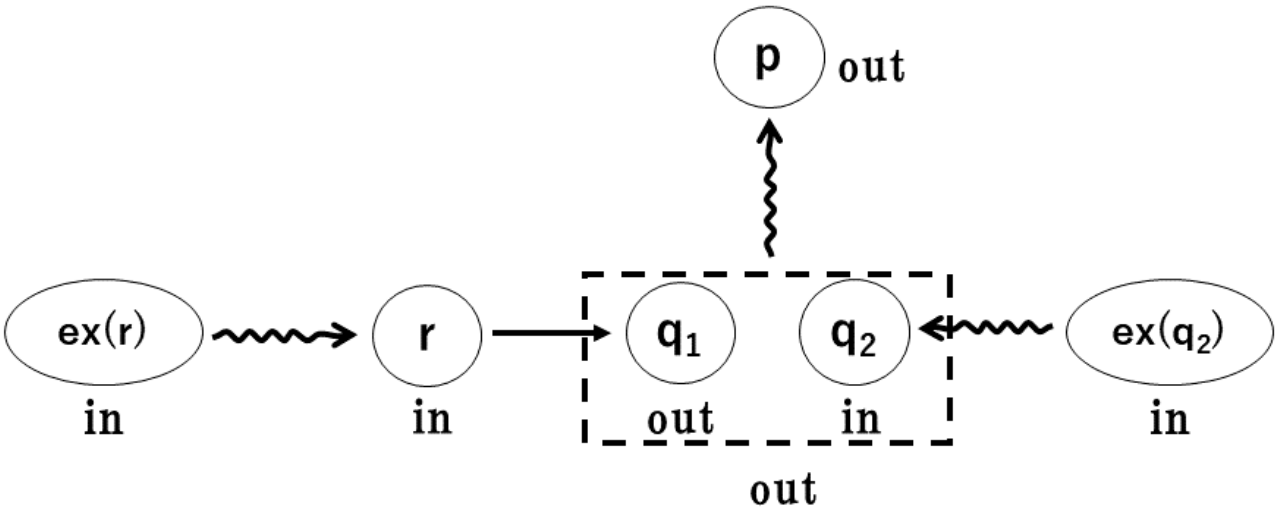
変換規則 (事実追加 existence)

```
q2<=.     % 事実
r<=.      % 事実
```



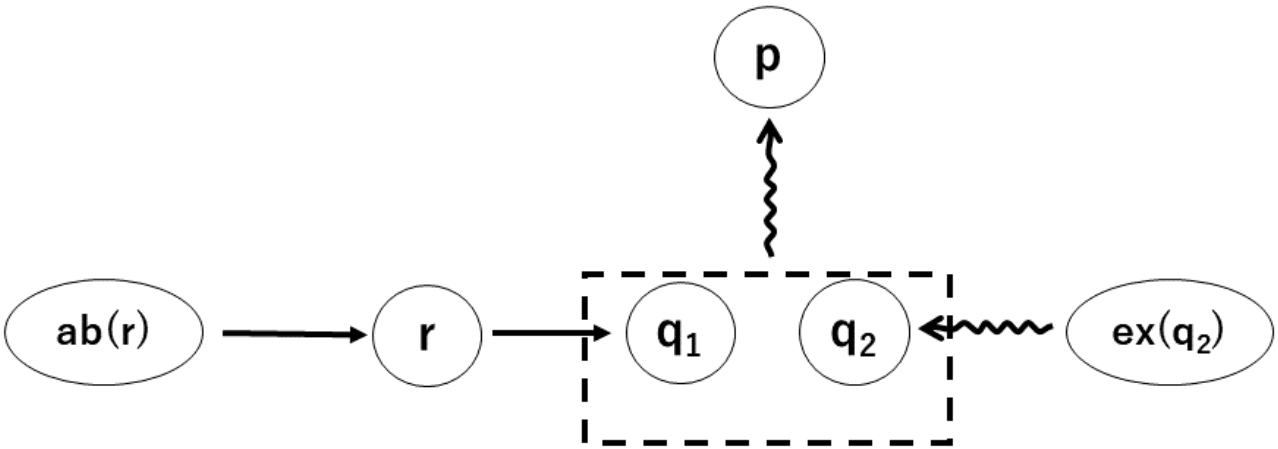
変換例 (全体)

```
p <= q1,q2.
exception(q1,r).
q2<=.
r<=.
```



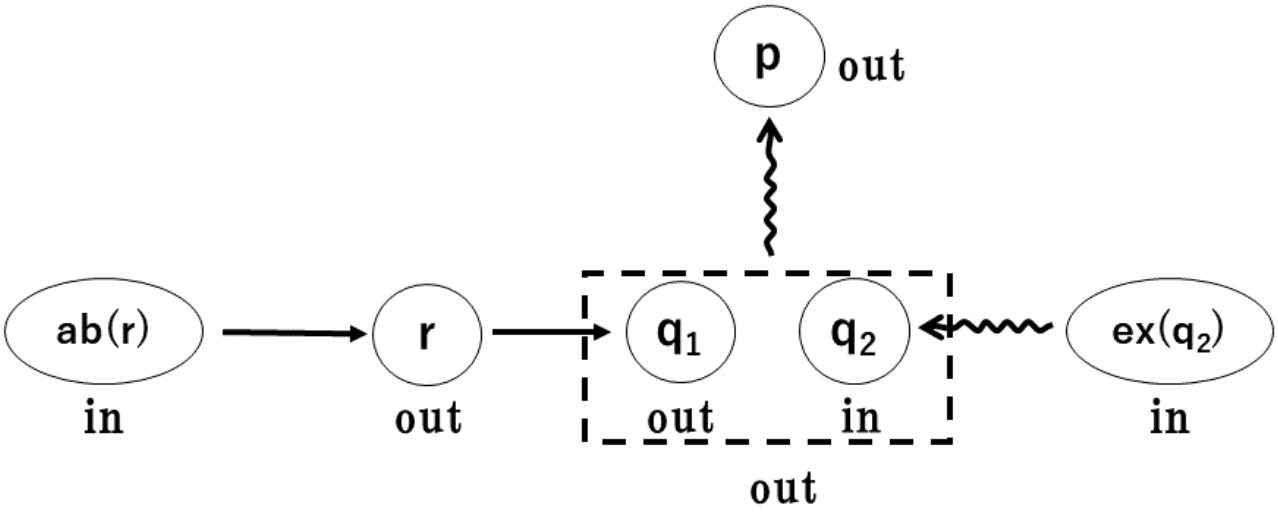
変換規則 (追加 absence)

```
p <= q1,q2.    % 規則
exception(q1,r).  % 例外
q2<=.    % 事実
```



変換全体

```
p <= q1,q2.    % 規則
exception(q1,r).  % 例外
q2<=.    % 事実
```



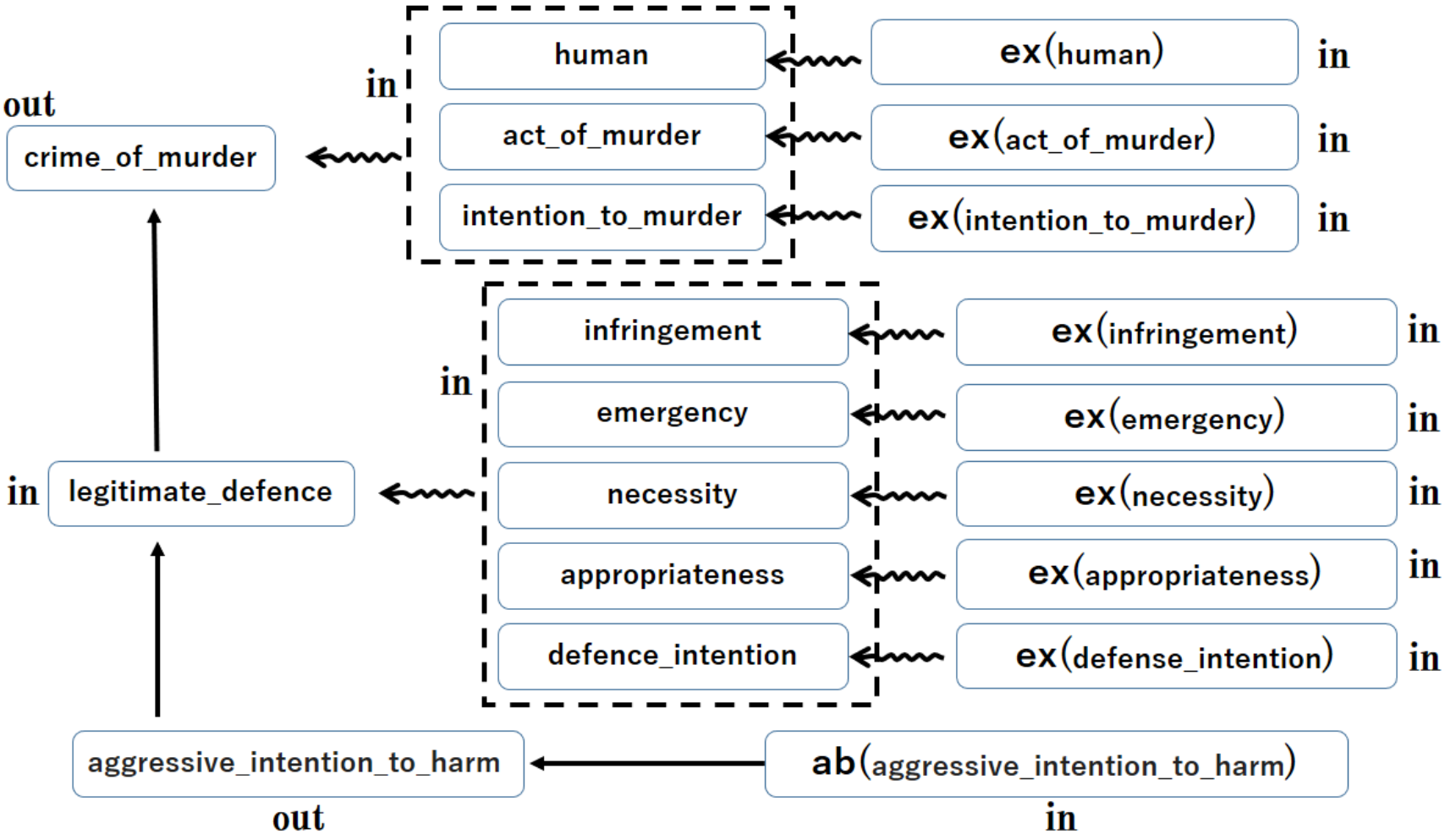
PROLEG 記述例 (規則と事実)

```
% rules regarding crime_of_murder
crime_of_murder <= human, act_of_murder, intention_to_murder.
exception(crime_of_murder,legitimate_defense).

legitimate_defense <=
  infringement, emergency, necessity, appropriatenessx,
  defense_intention.
exception(legitimate_defense,aggressive_intention_to_harm).

% facts
human <=.
act_of_murder <=.                    necessity <=.
intention_to_murder <=.              appropriateness <=.
infringement  <=.                    defense_intention <=.
emergency <=.
```

BAF の例


human
ex(human)
in
in
out
crime_of_murder
act_of_murder
ex(act_of_murder)
in
intention_to_murder
ex(intention_to_murder)
in
infringement
ex(infringement)
in
in
emergency
ex(emergency)
in
in
legitimate_defence
necessity
ex(necessity)
in
appropriateness
ex(appropriateness)
in
defence_intention
ex(defense_intention)
in
aggressive_intention_to_harm
ab(aggressive_intention_to_harm)
out
in

PROLEG 記述例 (規則と事実)

```
% rules regarding crime_of_murder
crime_of_murder <= human, act_of_murder, intention_to_murder.
exception(crime_of_murder,legitimate_defense).

legitimate_defense <=
  infringement, emergency, necessity, appropriateness,
  defense_intention.
exception(legitimate_defense,aggressive_intention_to_harm).
```

BAF 上の推論 - 必要な証拠の同定 (1)

条文に対する PROLEG 記述を BAF に変換し，その上で論証 A を in/out にするために必要な証拠を同定する．
以下を再帰的に実行する．

BAF 上の推論 - 必要な証拠の同定 (2)

(1) $\mathcal{L}(A) = in$ にするための条件
以下の双方が満たされる

- (attack condition) $\forall\ B$; $\mathrm{att}(B, A)$ に対して $\mathcal{L}(B) = out$ にする．そのような B が存在しない場合，この条件はすでに満たされている．
- (support condition) $\exists\ \mathbf{A}$; $\mathrm{sup}(\mathbf{A}, A)$ に対して $\mathcal{L}(\mathbf{A}) = in$ とする．そのような $\mathbf{A}$ が存在しない場合，$ex(A)$　および $\mathrm{sup}(\{ex(A)\}, A)$ を追加する．

BAF 上の推論 - 必要な証拠の同定 (2)

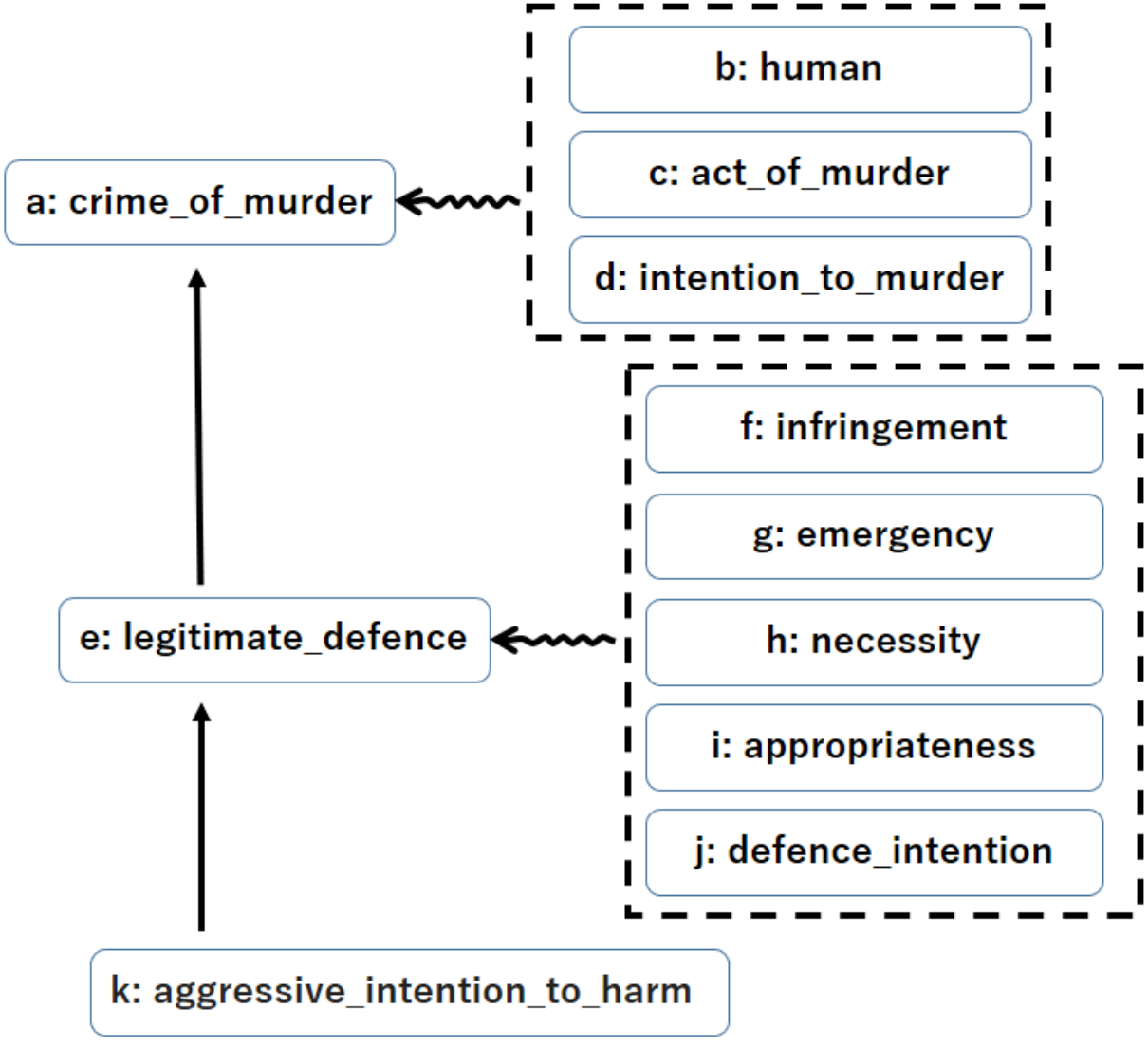
(2) $\mathcal{L}(A) = \mathit{out}$ にするための条件
以下のいずれかが満たされる

- (attack condition) $\exists\ B$; $\mathrm{att}(B, A)$ に対して $\mathcal{L}(B) = \mathit{in}$ にする．そのような B が存在しない場合，この条件は満たされない．
- (support condition) $\forall\ \mathbf{A}$; $\mathrm{sup}(\mathbf{A}, A)$ に対して $\mathcal{L}(\mathbf{A}) = \mathit{out}$ とする．そのような $\mathbf{A}$ が存在しない場合，この条件は満たされない．

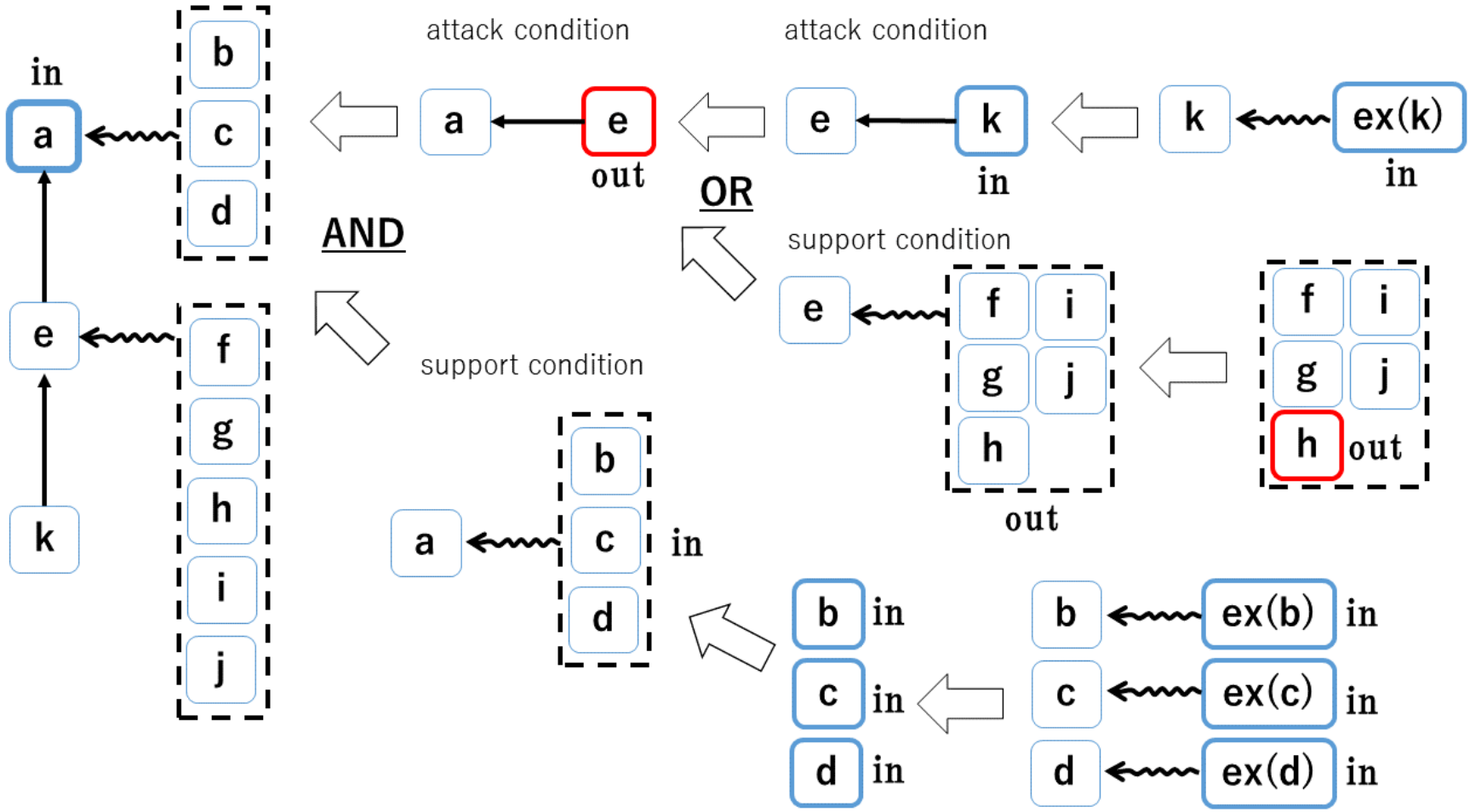
BAF 上の推論 - 必要な証拠の同定 (4)

このとき，追加された論証の集合が A を in/out にするために必要な論証となる．

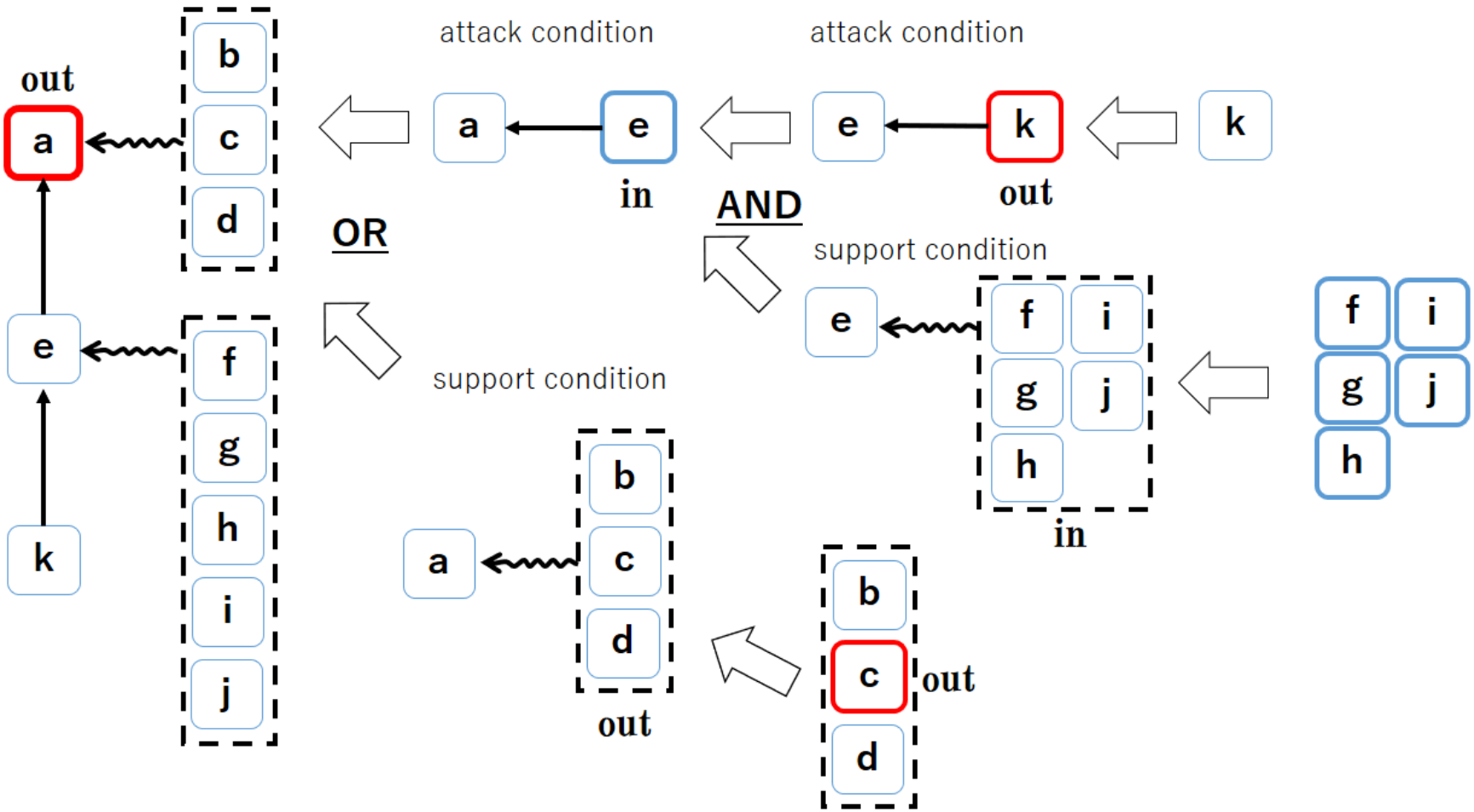
必要な証拠の同定例 (1)


b: human
c: act_of_murder
d: intention_to_murder
a: crime_of_murder
f: infringement
g: emergency
e: legitimate_defence
h: necessity
i: appropriateness
j: defence_intention
k: aggressive_intention_to_harm

必要な証拠の同定例 (2)

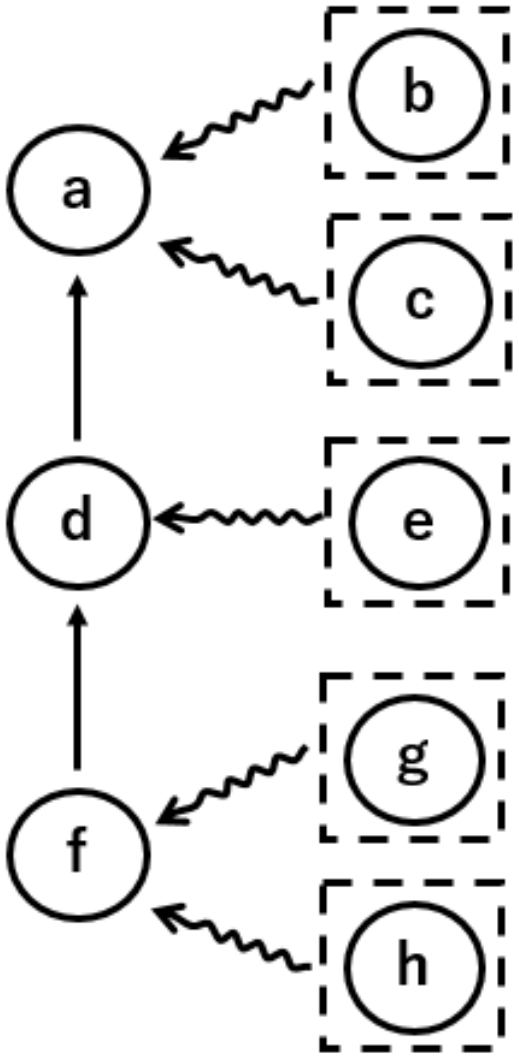

in
a
b
c
d
e
f
g
h
i
j
k
AND
attack condition
a
e
out
OR
attack condition
e
k
in
k
ex(k)
in
support condition
e
f
i
g
j
h
out
f
i
g
j
h
out
support condition
a
b
c
d
in
b in
c in
d in
b
ex(b)
in
c
ex(c)
in
d
ex(d)
in

必要な証拠の同定例 (3)


out
a
b
c
d
e
f
g
h
i
j
k
OR
attack condition
a
e
in
attack condition
e
k
out
k
AND
support condition
e
f
g
h
i
j
in
f
g
h
i
j
support condition
a
b
c
d
out
b
c
d
out

BAF 上の推論 - 必要な証拠の同定

$\mathcal{L}(a) = in$ にするための複数の解答が得られる場合



成果と今後の課題

- 成果
  - 非循環な PROLEG プログラム P における解集合と P を変換して得られる BAF $baf(P)$ における受理集合の一致することを証明
- 今後の課題
  - 循環な PROLEG からの変換
  - 標準論理プログラムからの変換

議論過程の可視化システム

1. PROLEG ブロック図による表示と編集
2. PROLEG から双極議論フレームワーク (BAF) への変換
3. 予測 AF にもとづく対話モデルの実行過程の表示

予測 AF にもとづく対話モデルの上での戦略について

1. エージェントが自分自身の知識と相手の知識の予測をもつモデルの提案
2. AF によってそれらを表現し説得対話モデルを構築して実装
3. 実験をおこない、説得成功できる戦略について考察
4. 議論の表示インタフェースを作成

不誠実な論証を含む説得対話

- 対話エージェントは自分自身の AF と相手の予測 AF の二つをもつ
- 議題を相手に納得させる（相手の AF 内のラベルを in にする）ように対話をすすめる
- 自分の知っていること（自分の AF 内にある論証）のみを発言できる
- 自分の信じていないこと（自分の AF 内のラベルを out にする）も発言する

Motivated Example (1)

Alice's knowledge

Charlie is strict.
Charlie is generous.
I like a strict professor.

Bob's knowledge

I don't like a strict professor.
I like a generous professor.

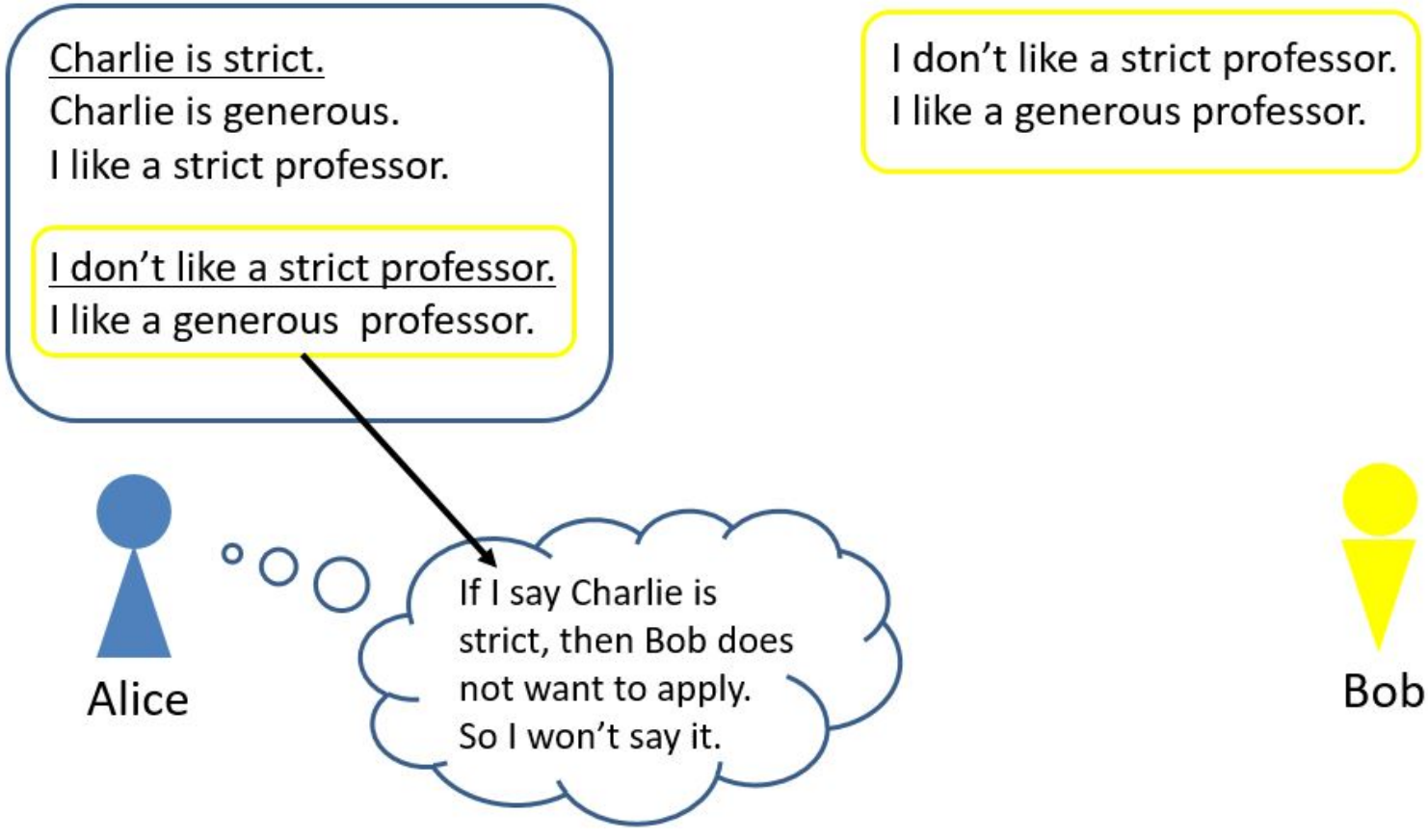
Bob's knowledge

I don't like a strict professor.
I like a generous professor.

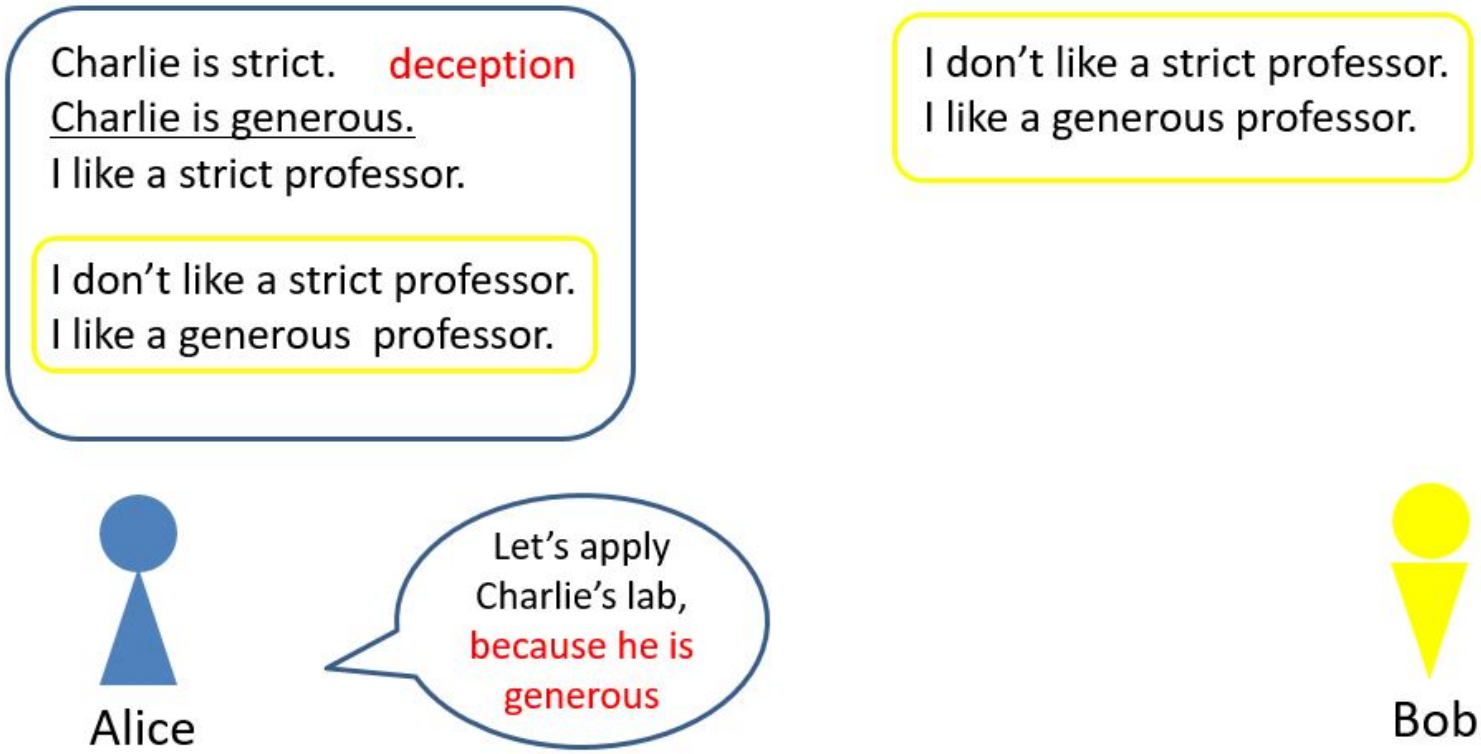
Alice

Bob

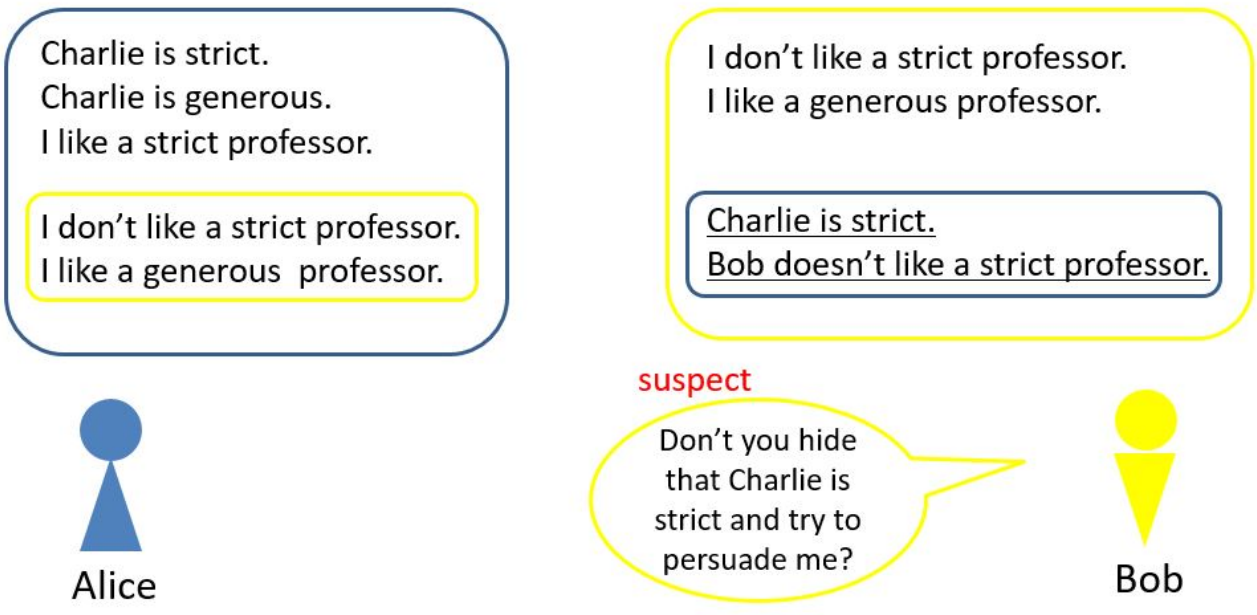
Motivated Example (2)



Motivated Example (3)

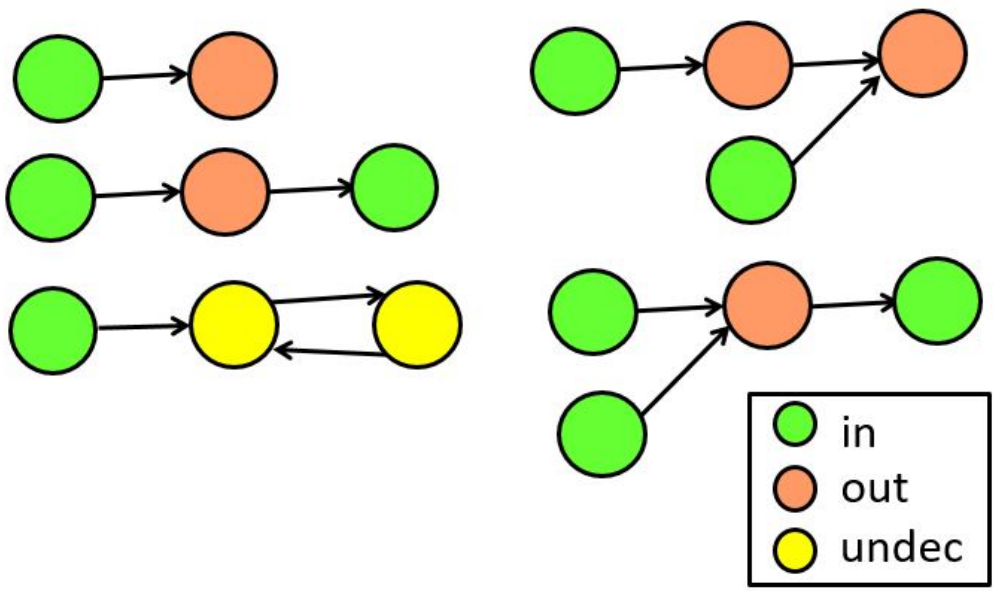


Motivated Example (4)



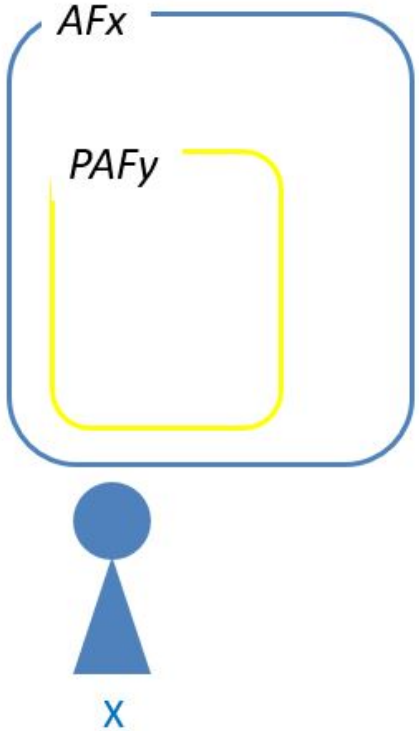
(grounded) Labelling

- A function $\mathcal{L}$: from a set of arguments to $\{in, out, undec\}$
  - $\mathcal{L}(A) = in$ iff for any A' that attacks A, $\mathcal{L}(A')$ is out
  - $\mathcal{L}(A) = out$ iff there exists some A' that attacks A, s.t. $\mathcal{L}(A) = in$
  - $\mathcal{L}(A) = undec$ otherwise
- An agent's beliefs: Arguments with label in



An Agent's AFs

- Each agent X has
  - her own AF ($\mathcal{AF}_X$)
  - prediction of her opponent Y's AF ($\mathcal{PAF}_Y$)
  - $\overline{\mathcal{P}\mathcal{AF}_Y}$ is a sub-argument of $\mathcal{AF}_X$
- Each agent X gives an argument A based on these AFs

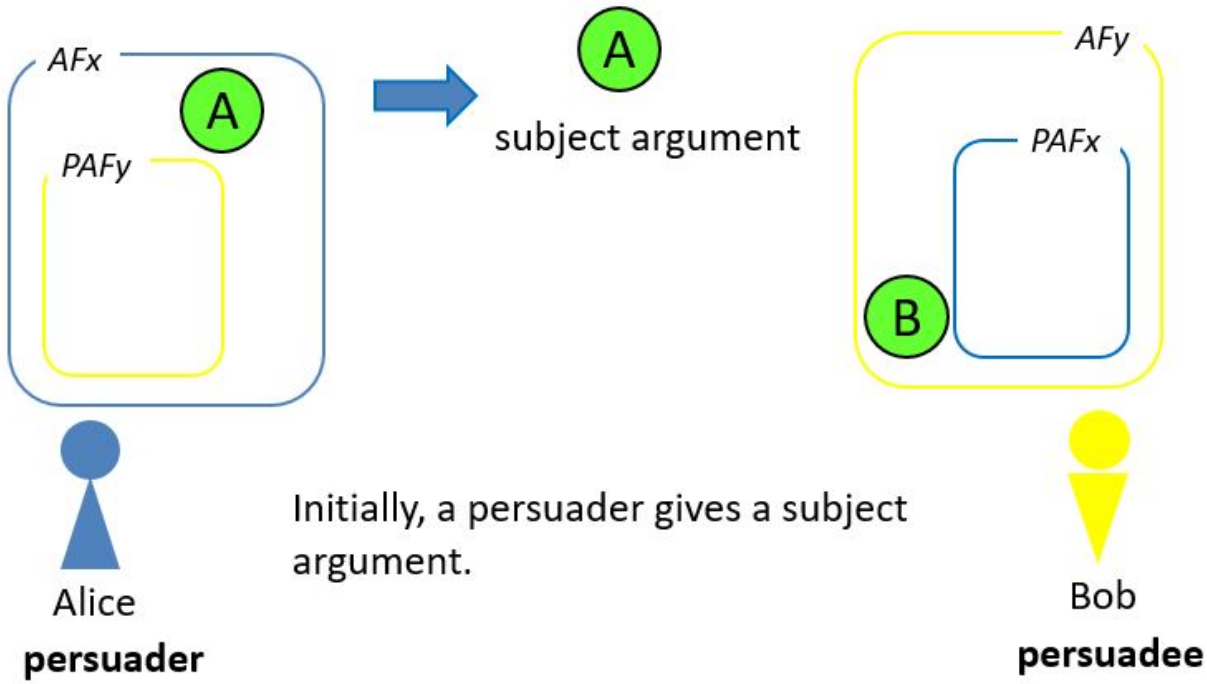


Initial State

A dialogue proceeds like this.

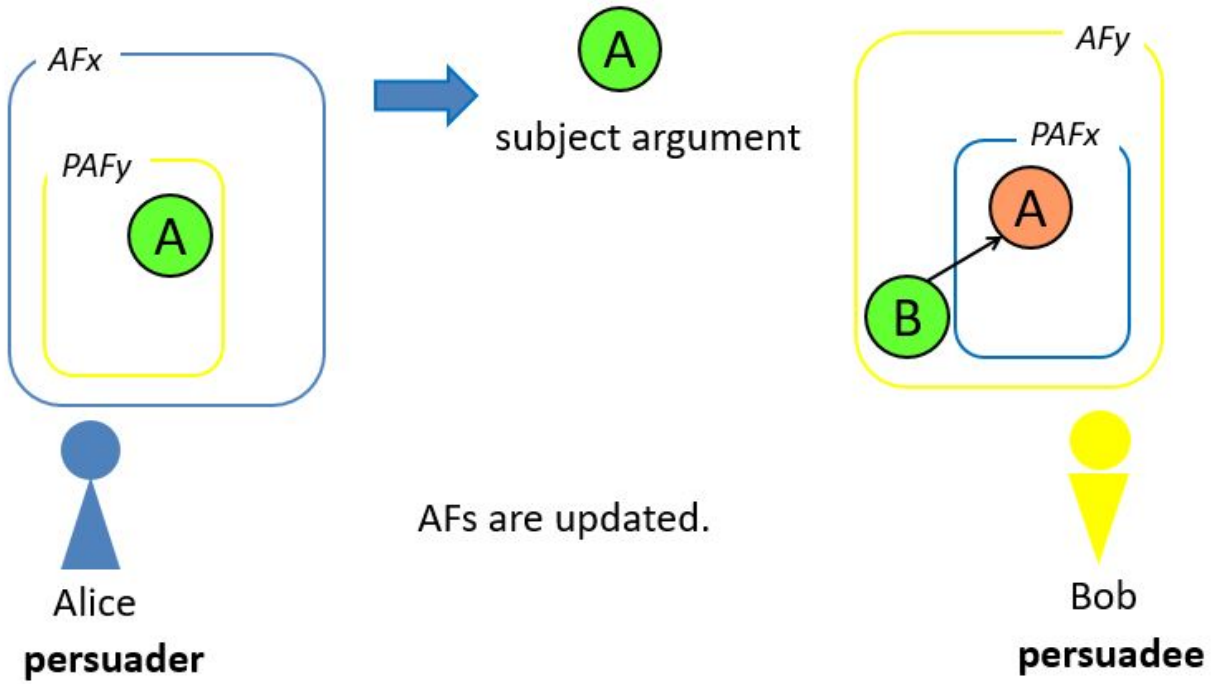
Initial State

A dialogue proceeds like this.



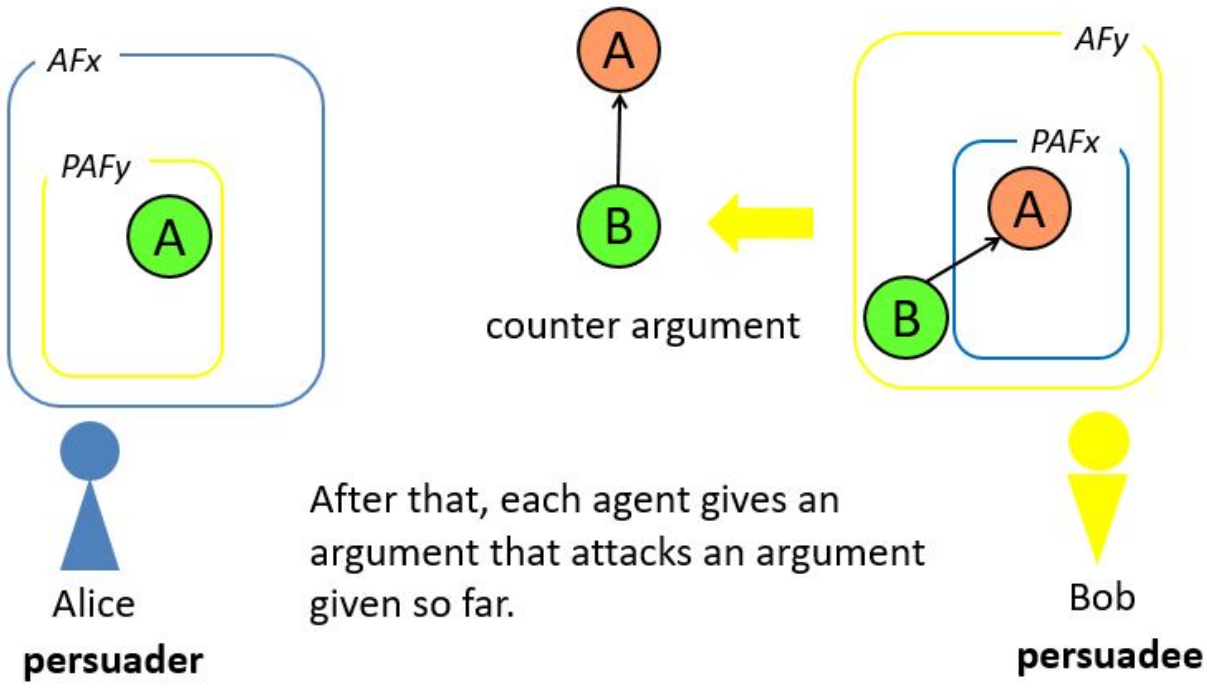
Updates

A dialogue proceeds like this.



Counter Arguments

A dialogue proceeds like this.



AFs are updated every time an argument is given and their labels are changed.

Act of an Argument

Act

- assert: simple counter argument
- suspect: pointing out deception
- excuse: excuse for the pointing
- pass: passing her turn

If suspect is given, then the opponent should give an excuse immediately.

議論過程の可視化システム

1. PROLEG ブロック図による表示と編集
2. PROLEG から双極議論フレームワーク (BAF) への変換
3. 予測 AF にもとづく対話モデルの実行過程の表示
4. その他

今後の予定

- インタフェース開発
  - ブロック図インタフェースの拡張（11 月から外注？）
- 理論研究
  - NLP から BAF への変換
  - ADF など受理条件の論理式表現との対応づけ
- AF の拡張
  - AF の分割
  - modularity の導入と Baysian Net を利用した理由づけ
  - 仮説推論の導入？