議論過程の可視化システム

高橋 和子

関西学院大学

2018 年 8 月 11 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

概要

- グラフ上の編集機能の向上
- 表示機能の改良と拡張

PROLEG 実行過程のブロック図

- 佐藤らによって開発されたブロック図の機能拡張
- PROLEG の実行結果を、各ノードが論証，エッジが攻撃または支持とするグラフ表現 (木構造)
- 各論証は正しいと証明されたら o, 証明されなかったら x という評価値をもつ
- 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
  exception(H,E).      % 例外
  ```
- 意味：B1,...,Bn が成り立ち、かつ、E が成り立たないとき、H は成り立つ
- ```
  H<=.     % 事実
  ```
- 意味：H は成り立つ

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
- 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。
- ノードの構成要素（各論証）の評価値を o, x の二値から u を加えた三値に拡張 (真偽値が不明なものは保留 (否認) u として処理
- 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予定

- ノード編集において可能な述語のメニューの表示
- 注目ノードの評価に影響する事実をリストアップし、そこから指定できるようにする（注目ノードの評価値を変更するためにはどの事実が証明されればよいかを表示）
- or 関係にあるノードの表示の切替と、ユーザによる指定
- 論証に対する反論があるかどうかを尋ねる機能
- 事実が成り立つか (証拠があるか) どうかを仮説とした推論機能 (?)
- 戦略の決定支援 (?)

PROLEG から双極議論フレームワーク (BAF) への変換

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

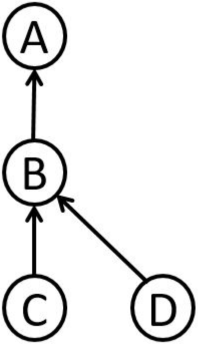
議論フレームワーク (Argumentation Framework, AF)

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

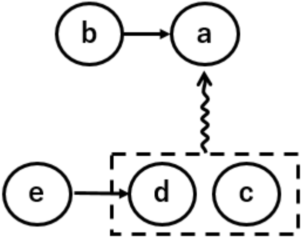
$baf = \langle AR, ATT, SUP \rangle$

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

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。

AR is a set, $ATT \subseteq AR \times AR$, $SUP \subseteq (2^{AR} \backslash \emptyset) \times AR$

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
- PROLEG プログラム P に対する解集合 M：
  M が集合 $P^M = \{H \Leftarrow B_1, \ldots, B_n \in \mathcal{R} | \forall E \in \mathcal{E}, \text{ if } head(E) = H \text{ then } body(E) \not\subseteq M\}$ の最小モデルである．[Satoh11]

BAF の意味論

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

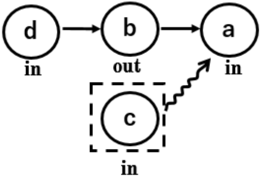
BAF の受理集合: $\{A \mid \mathcal{L}(A) = in\}$
対象とする日本の民法、刑法は非循環な定義になっているべきなので、非循環なプログラムのみを対象とする

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

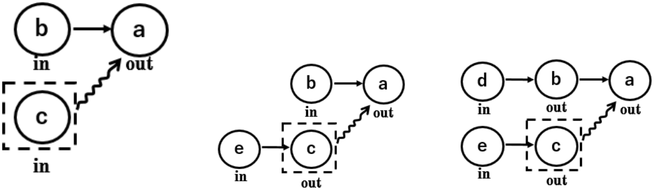
- $\forall B \in AR(\neg \mathrm{att}(B, A) \wedge) \neg \forall \mathbf{A} \subseteq AR(\mathrm{sup}(\mathbf{A}, A))$ ならば $\mathcal{L}(A) = in$.
- $(\forall B \in AR, \mathrm{att}(B, A) \Rightarrow \mathcal{L}(B) = out) \wedge (\exists \mathbf{A} \subseteq AR, \mathrm{sup}(\mathbf{A}, A) \wedge \mathcal{L}(\mathbf{A})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

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



支持がないと in ではない
攻撃が支持よりも強い

変換規則

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

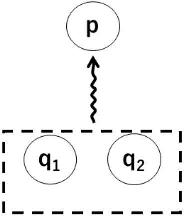
- $Atom = \bigcup_{R \in \mathcal{R}}(head(R) \cup body(R)) \cup \bigcup_{E \in \mathcal{E}}(head(E) \cup body(E))$
- $Rule = \{(body(R), head(R)) \mid R \in \mathcal{R} \wedge body(R) \neq \emptyset\}$
- $Exc = \{(B, H) \mid \mathrm{exception}(H, B) \in \mathcal{E}\}$
- $Existence = \{H \mid H \Leftarrow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

変換例:PROLEG プログラム

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

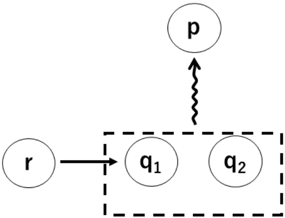
変換規則 (規則)

```
p <= q1,q2.    % 規則
```



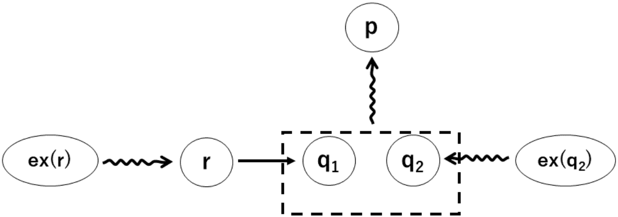
変換規則 (例外)

```
exception(q1,r).  % 例外
```



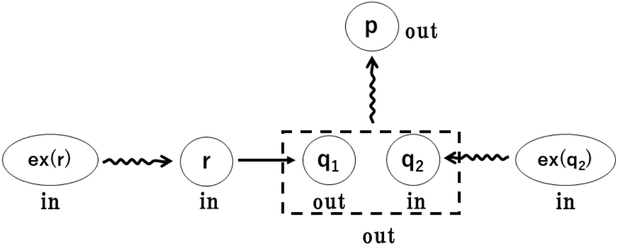
変換規則 (事実)

```
q2<=.     % 事実
r<=.      % 事実
```



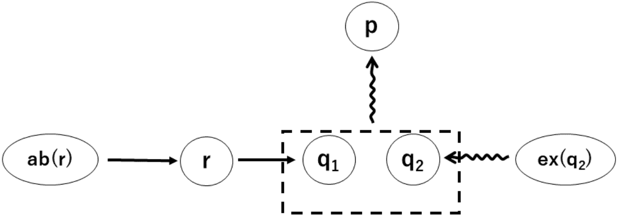
変換例 (全体)

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



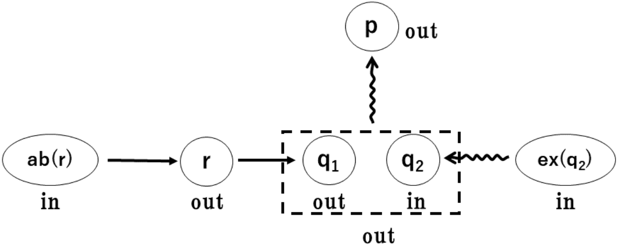
変換規則 (追加 absence)

```
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
```



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

予測 AF にもとづく対話モデルの実行過程

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