論理型言語PROLEGに対応する双極議論フレームワークの意味論について

川崎 樹　森口 草介　高橋 和子

論理型プログラム PROLEG から双極議論フレームワークへの変換について述べる．PROLEG は日本における法律要件を記述するために開発されたシステムであり，民法や刑法を記述するには適しているが，それを実際の案件に適用したときにおこる推論過程や論証のやりとりなどは簡単にはわからない．これらを明確に表現するために PROLEG プログラムを，論証および論証間の攻撃関係，支持関係によって議論を表現するシステムである双極議論フレームワークに変換する．本稿では，非循環な PROLEG プログラムを対象として，変換によって得られる双極議論フレームワークに意味論を与え，この変換が解集合で与えられた PROLEG の意味論を保持することを示す．また，循環を含む場合についてのこの手法の拡張について議論する．

1　はじめに

ここ 30 年ほど計算機上の議論学 (argumentation) に関する研究は非常にさかんに行われている [4]．議論学の考え方を使うことで，論理的なモデルが提示でき，結論だけでなく，結論を出すまでの経過や各論証の因果関係についても明確に示すことができるため，マルチエージェント環境での合意形成，対話システム，非単調推論といった人工知能の分野だけでなく，システム設計における信頼性検証などソフトウェア工学の分野でも議論学が応用されている．法的推論も議論学の有用な応用の一つとして期待されている分野である．

裁判におけるやりとりは一種の議論と考えられるが，判決を出す過程は人間が行っており，システム化されていないのが現状である．法律とその適用をシステム化し計算によって法的推論を行うために PROLEG システムが開発された [5].

法には効果を発揮するための前提条件があり，前提条件を満たすための事実が必要である．それぞれを構成要件，(要件) 事実という．また，法には違法性を否定する阻却事由など例外的なルールがあることもある．法が効果を発揮するための前提条件の一つに例外的なルールが成立しないことがあげられる．PROLEG ではこの構成要件，事実，例外的なルールの関係を論理プログラムで表現する．

PROLEG の記述言語は Prolog を拡張したもので，その記述は 2 種類の節からなる．一つは一般的な規則であり，もう一つは例外と呼ばれる特殊な規則である．例えば，殺人罪を PROLEG で記述することを考える．殺人罪の構成要件として「自然人である」,「殺人行為がある」「殺意がある」の三つがあるとして，「自然人であり，殺人行為があり，殺意があったならば殺人罪が成り立つ」ことを PROLEG で記述すると次のように書くことができる．

殺人罪 <= 自然人，殺人行為，殺意．

また，殺人罪に対する例外として正当防衛を考えると,「正当防衛ならば殺人罪でない」ことは PROLEG で以下のように記述できる．

`exception(殺人罪，正当防衛)`.

この 2 節から推論を行う．例えば三つの構成要件について全ての事実が確認できれば，殺人罪が成立する．一つでも事実が確認できなければ殺人罪は成立しない．また，正当防衛が成立するなら，たとえ殺人罪

On the semantics of the Bipolar Argumentation Framework corresponding to logical language PROLEG.

Tatsuki Kawasaki, Sosuke Moriguchi, Kazuko Takahashi, 関西学院大学, Kwansei Gakuin University.

の構成要件の全て事実が確認できていても殺人罪は成立しない．このように推論を行い結論を導く．

訴訟現場において，ある法の適用もしくは不適用のプロセスを提示することは重要である．法曹は結果と理由は提示するが，構成要件や事実を用いて推論過程を示すことはない．そのため原告と被告の双方が判決に十全に納得しない可能性が考えられる．そのため，構成要件や事実の関わりを踏まえた推論過程を示す必要がある．その際，PROLEG の記述を提示しても，論理プログラムは法曹にはなじみのない記述法式である上に推論過程が明示されていないので，理解は容易ではない．

そこで，法的推論を一種の議論とみなせることを利用する．議論の過程や結論に至る理由を明確にする研究として，議論システムに基づく手法が多く提案されている [4]．Dung は抽象レベルで議論の仕組みをとらえた議論フレームワーク (Argumentation Framework, AF) を提唱した [3]．この枠組みでは論証および論証間における攻撃関係の二項組で定義され，どの論証が受理されたかを計算することで意味論が与えられている．その後この手法から多くの派生的研究が行われている．

双極議論フレームワーク (Bipolar Argumentation Framework, BAF) もその一つであり，Dung の議論フレームワークに攻撃関係だけではなく支持関係を追加した 3 項組で定義されている [1]．裁判においては攻撃あるいは支持となる論証が陽に現れるため，AF よりも BAF による表現が望ましい．

本研究の目標は，計算機システムや論理プログラミングになじみのない法曹に対して裁判における推論過程や議論の進行状況を明確にすることであり，この目標を達成するために，PROLEG の記述を BAF に変換するアプローチをとる．また，BAF における支持関係を論証間におけるものではなく，論証の集合と論証の間におけるものに拡張し，それに意味論を与える．BAF については Cayrol らが意味論を与えているが [2]，PROLEG に対応する BAF については既存の意味論を適用してもそのままでは両者の意味論は一致しない．本研究では BAF に新たに意味論を与えることで PROLEG プログラムから BAF へ意味を保存したまま変換することを可能にした．

本稿の構成は以下の通りである．まず，2 節で PROLEG について述べる．次に，3 節で我々が対象とする BAF を定義し，それに意味論を与える．その後，4 節で PROLEG プログラムから BAF への変換手法を述べる．5 節では刑法を記述した PROLEG プログラムが非循環であることを利用し，変換して得られた BAF が PROLEG プログラムの意味を保持していることを証明する．6 節で循環を含むプログラムの意味論について議論し，7 節で結論と今後の課題について述べる．

2 論理型言語 PROLEG

論理型言語 PROLEG は Prolog を拡張して開発された裁判支援を目的とする記述言語である [5]．ある法に対して，構成要件，事実，例外的なルールを組み合わせ記述することで，その法の真偽を判定する．構成要件は法が効果を発揮するための条件であり，事実もしくは他の法の成立が必要となる．例外的なルールとは法もしくは構成要件の成立を，例外的に否定する効果を持つ．例外的なルールが認められれば，他の要素によって成立していても例外が認められ不成立となる．佐藤らは，PROLEG を用いて記述されたプログラムを以下のように定義している．法と構成要件の関係を規則で表し，例外的なルールを例外で表した．

ここで，$H, B, B_1, \ldots, B_n$ はアトムである．

定義 1. PROLEG プログラム P は，$P = \langle \mathcal{R}, \mathcal{E} \rangle$ の 2 つ組で定義される．$\mathcal{R}$ は規則の有限集合，$\mathcal{E}$ は例外の有限集合である．

定義 2. P 中において，$H \Leftarrow B_1, \ldots, B_n$ の形式で記述された節を規則という．特に，$H \Leftarrow$ の形式で記述された規則を事実という．

定義 3. P 中において，exception(H, B) の形式で記述された節を例外という．

以下に例を示す..

例 1.

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

`p <= q1,q2` は規則である．`p <= q1,q2` では，`q1`，`q2` が真であるならば `p` は真となることを表す．`q2<=`，`r<=`は事実であり，真であることを表す．`exception(q1,r)` は例外である．`r` が真であるならば，その時に限り `q1` は必ず偽となることを表す．

以下では，P を可能な置換によって基礎化されたものとして扱う．

定義 4. 関数 *head*，*body* を以下のように定義する．規則 $R = H \Leftarrow B_1, \ldots, B_n$ について，$head(R) = H$，$body(R) = \{B_1, \ldots, B_n\}$ である．例外 $E = \text{exception}(H, B)$ について，$head(E) = H$，$body(E) = \{B\}$ である．

また，佐藤らは PROLEG プログラム P の解集合を以下のように定義した．

定義 5. P に対して解集合 M を次のように定義する．M が集合 $P^M = \{H \Leftarrow B_1, \ldots, B_n \in \mathcal{R} \mid \forall E \in \mathcal{E}, \text{if } head(E) = H \text{ then } body(E) \nsubseteq M\}$ の最小モデルである．

本稿ではさらに各アトムに対して，関連集合を定義する．さらに PROLEG プログラムに対してレベル関数を定義する．

定義 6. P 中の各アトム A に対して，関連集合 $r_P(A)$ を次のように定義する．$r_P(A) = \bigcup_{R \in \mathcal{R} \wedge head(R) = A} body(R) \cup \bigcup_{E \in \mathcal{E} \wedge head(E) = A} body(E)$

定義 7. レベル関数 l_P を，P 中のアトムから非負整数への関数として以下のように定義する．

$$l_P(A) = \begin{cases} 1 & (r_P(A) = \emptyset) \\ max_{B \in r_P(A)} l_P(B) + 1 & (\text{otherwise}) \end{cases}$$

l_P が一意に定まるとき，P を非循環であるという．まず初めに非循環な PROLEG プログラムを対象にする．よって，$l_P(A) = n$ のとき，A のレベルは n である．

3 双極議論フレームワーク

Cayrol らは，抽象議論フレームワークを拡張し，論証と論証間の二項関係として，攻撃の支持の 2 種類を用いた 3 つ組で定義される双極議論フレームワーク (BAF) を提唱した [2]．本研究では支持関係を拡張し，論証と論証の集合の間の二項関係として定義する．

定義 8. 双極議論フレームワーク BAF は $baf = \langle AR, ATT, SUP \rangle$ の組として定義される．AR は論証の有限集合であり，ATT，SUP はそれぞれ $ATT \subseteq AR \times AR$，$SUP \subseteq (2^{AR} \setminus \emptyset) \times AR$ で定義される集合である．$(B, A) \in ATT$ を $\text{att}(B, A)$，$(\mathbf{A}, A) \in SUP$ を $\text{sup}(\mathbf{A}, A)$ と記述する．

定義 9. ある論証 $A \in AR$ に対して，A の関連集合 rel_A を $rel_A = \{B \mid \text{att}(B, A)\} \cup \{B \mid B \in \mathbf{A} \wedge \text{sup}(\mathbf{A}, A)\}$ で定義する．

定義 10. 高さ関数 h を，$baf = \langle AR, ATT, SUP \rangle$ の AR から非負整数への関数として以下のように定義する．

$$h(A) = \begin{cases} 0 & (rel_A = \emptyset) \\ max_{B \in rel_A} h(B) + 1 & (\text{otherwise}) \end{cases}$$

ラベリングは BAF における各論証の受理性を表す．ラベリングにより各論証の受理性は異なるため，ラベリングは BAF に意味を与えることに他ならない．ここでは，PROLEG プログラムから BAF へ変換するため，PROLEG プログラムの意味を保持するためのラベリングを与える．以下のようにラベリングを定義する．

定義 11. $baf = \langle AR, ATT, SUP \rangle$ 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．また，$\mathbf{A} \in (2^{AR} \setminus \emptyset)$ に対して，$\forall A \in \mathbf{A}, \mathcal{L}(A) = in$ のときに $\mathcal{L}(\mathbf{A}) = in$，そうでなければ $\mathcal{L}(\mathbf{A}) = out$ とする．

また，完全ラベリングを以下のように定義する．

定義 12. $baf = \langle AR, ATT, SUP \rangle$ において，任意の $A \in AR$ に対するラベリング $\mathcal{L}$ が以下を満たすならばそのラベリングを完全ラベリングという．

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

任意の非循環な BAF には，唯一つの完全ラベリングが存在する．また，$\{A \mid \mathcal{L}(A) = in\}$ である集合を BAF の受理集合という．

例 2. 次の BAF を考える．

$\langle \{a, b, c, d\}, \{(b, a), (d, b)\}, \{(\{c\}, a)\} \rangle$

この BAF の完全ラベリングを図 1 に示す．

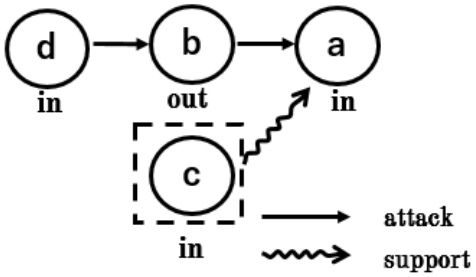

図 1　完全ラベリング付き BAF

4 PROLEG プログラムから双極議論フレームワークへの変換

PROLEG プログラム P を BAF に，P の解集合と BAF の受理集合が一致するように変換する．この変換により P 中のアトム，規則，例外はそれぞれ論証，支持関係，攻撃関係に変換される．さらに，P 中に明示的には現れない二つの論証を BAF に追加する．1 つは P 中のあるアトムに対して，そのアトムが成立するか否かが，P 中に明記されていないことを示す不在論証である．P の記述の中で一度も頭部として現れないアトムは，成立するか否かが明記されておらず，解集合に含まれることはない．一方で，BAF では攻撃も支持もされていない論証のラベルは *in* である．この差を埋めるため，不在論証を加えそのアトムを攻撃する．あるアトム A に対する不在論証を ab(A) と書く．

もう一つは，事実の存在を示す存在論証である．事実は体部にアトムを持っていないが，BAF では支持は二項関係である．そのため存在論証を加えそのアトムを支持する．事実として記述されたアトム A に対して，存在論証を ex(A) と書く．

定義 13. PROLEG プログラム $\langle \mathcal{R}, \mathcal{E} \rangle$ から $baf = \langle AR, ATT, SUP \rangle$ への変換を以下のように定義する．

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

例 3. 次の PROLEG プログラムを考える．

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

この PROLEG プログラムを変換して得られた BAF は次のようになる．

$$\langle \{p, q_1, q_2, r, \text{ex}(q_2), \text{ex}(r)\}, \{(r, q_1)\}, \{(\{q_1, q_2\}, p), (\{\text{ex}(q_2)\}, q_2), (\{\text{ex}(r)\}, r)\} \rangle$$

論証 q_1，q_2 は集合で p を支持している．またこの例では q_2，r に対してそれぞれ存在論証が作成されている．図 2 にこの BAF のラベリング付きグラフ表現を示す．

5 意味論を保持していることの証明

PROLEG プログラムを変換して得られた BAF は PROLEG プログラムの意味論を保持している．

定理 1. baf を PROLEG プログラム P を変換して得られた BAF とする．また，M を P の解集合とし，$\mathcal{L}$ を baf の完全ラベリングとする．このとき，P に出現する任意のアトム H について，$\mathcal{L} = in$ ならばそのときに限り $H \in M$ である．

これの証明を示す．

Proof. アトムのレベル n に関する帰納法による．baf において高さ 0 の論証は不在論証もしくは存在論証であるから，明らかに P のレベルは baf における高さとして保持される．

$n = 1$ の時，H は事実もしくは一度も頭部として現れていないアトムである．もし H が事実であるならば，$H \in M$ である．この時，事実は sup($\{\text{ex}(H)\}, H$) に変換されており (かつ他の論証からの攻撃関係はない),

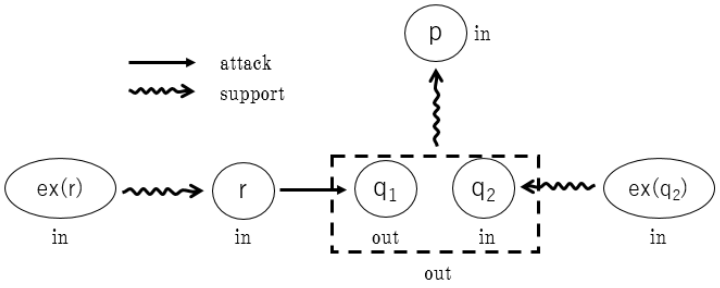

図 2　変換して得られた完全ラベリング付き BAF

$\mathcal{L}(\text{ex}(H)) = in$ であるから $\mathcal{L}(H) = in$ である．もし H が一度も頭部として現れていないアトムであるなら，exception(ab(H), B) に変換されており，$\mathcal{L}(\text{ab}(H)) = in$ であるから $\mathcal{L}(H) = out$ である．よって，$\mathcal{L}(H) = in$ であるならそのときに限り $H \in M$ である．

$n \geq 2$ について，任意の $k < n$ で定理を満たすと仮定する．つまり，レベル k の任意のアトム H' について $H' \in M$ であるなら $\mathcal{L} = in$ であるとする．H のレベルは 1 より大きいので H は規則または例外，あるいはその両方に出現している．

初めに $H \in M$ の時 $\mathcal{L}(H) = in$ であることを示す．M は最小モデルなので，全ての m について $B_m \in M$ であるような規則 $H \Leftarrow B_1, \ldots, B_m$ が存在し，$B \in M$ であるような例外 exception(H, B) が存在しない．*baf* において規則は sup($\{B_1, \ldots, B_m\}, H$) に変換される．$B_1, \ldots, B_m$ のレベルは n より小さいので，仮定より，全ての m について $\mathcal{L}(B_m) = in$ である．また例外は att(B, H) に変換され，$B \notin M$ であり，B のレベルは n より小さいので $\mathcal{L}(B) \neq in$ である．よって $\mathcal{L}(H) = in$ である．

次に，$\mathcal{L}(H) = in$ であるならば $H \in M$ であることを示す．H は他のいくつかの論証から支持または攻撃されている，あるいは支持も攻撃もされている．よって $\mathcal{L}(H) = in$ であるならば，任意の H を攻撃している論証 B について $\mathcal{L}(B) \neq in$ である．この攻撃関係は例外 exception(H, B) を変換して得られた関係であり，仮定より $B \notin M$ である．また，全ての m について $\mathcal{L}(B_m) = in$ であるような H を支持している論証集合が存在している．このような支持関係は規則 $H \Leftarrow B_1, \ldots, B_m.$ を変換して得られた関係であり，$B_1, \ldots, B_m$ のレベルは n より小さいので，仮定より $B_1, \ldots, B_m \in M$ である．M は最小モデルであるので，$H \in M$ である．

従って，全てのレベル n について，$\mathcal{L}(H) = in$ であるならばそのときに限り $H \in M$ である．　□

6　循環を含むプログラムへの拡張についての考察

ここまでは，非循環な PROLEG プログラムのみを取り上げた．これは PROLEG が対象とする日本の法，特に刑法では適切に構成されているならば規則同士が衝突するようなことがないためである．しかし，PROLEG プログラムは，循環を含む記述も許す．たとえば [6] では以下のような例が記述されている．

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

q1 と q2 がそれぞれに対して例外を主張しており，一方を認めるともう一方が認められない記述となっている．この PROLEG プログラムの解集合は，{*q1*} および {*q2*} の二つである．

このプログラムを 4 節の手法によって BAF に変換すると，$\langle \{q1, q2, \text{ex}(q1), \text{ex}(q2)\}, \{(q1, q2), (q2, q1)\}, \{(\{\text{ex}(q1)\}, q1), (\{\text{ex}(q2)\}, q2)\} \rangle$ となる．この BAF における完全ラベリングは二つ考えられ，一方は *q2* だけを *out* にするもの，もう一方は *q1* だけを *out* にするものである．先ほど挙げた解集

合とこのラベリングはそれぞれ対応している．

非循環な場合の証明はレベル関数および高さ関数を用いて行った．しかし循環が存在するとレベル関数および高さ関数が定義できない．この問題は，プログラムや BAF に対する関数をどのような意味に対しても単一に定義しようとしたために発生する問題である．そこで，解集合やラベリングに応じた関数を定義し，その上で非循環な場合に似た証明を行うことを考えている．証明の完成は今後の課題だが，成功すれば一般の標準論理プログラムを意味的に等価な BAF に変換する手法とその正当性を与えることができる．

ただし，もう一点，PROLEG 特有の意味論に関する問題が存在する．以下のプログラムを考える．

```
q1 <= q2.
q2 <= q1.
```

このプログラムは q1 と q2 が互いに導出し合う，ある種の循環論法になっている．PROLEG では解集合を最小モデルと規定しているため，このプログラムの解集合は空集合唯一つに定まる．一方，変換後の BAF では二つの完全ラベリングがあり，一つは空集合と対応するが，もう一つは全体集合と対応する．

一般の標準論理プログラムにおいては，全体集合もやはり解集合と認められる．しかし，PROLEG は法的推論を目的としているため，根拠が明確でない循環論法を認めない．この差を埋めるため，BAF における完全ラベリングより制約の強い意味論を与え，対応を考える必要がある．ただし，上の問題は他から支持されていない循環する支持関係のみが原因となるため，既存の完全ラベリングから大きく変更を加える必要はないと考えている．

7 おわりに

非循環な PROLEG プログラムから BAF への変換を行った．変換した BAF に対して，PROLEG プログラムの解集合を保持する意味論を与え，さらに変換して得られた BAF が解集合を保持していることを証明した．また，循環する PROLEG プログラムに対する考察を行った．今後は厳密に PROLEG と対応する BAF の意味論を与え，証明を行う．

謝辞

本研究は JSPS 科研費 JP17H06103 の助成を受けたものです．

参考文献

[1] Amgoud, L., Cayrol, C., Lagasquie-Schiex, M. and Livet, P.: On bipolarity in argumentation frameworks. *International Journal of Intelligent Systems*, Volume 23,1062-1093 (2008).

[2] Cayrol, C. and Lagasquie-Schiex, M.: Bipolarity in argumentation graphs: Towards a better understanding. *International Journal of Approximate Reasoning*, Volume 54, 876-899 (2013).

[3] Dung, P. M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, Volume 77, 321-357 (1995)

[4] Rahwan,I. and Simari,G.(eds.): *Argumentation in Artificial Intelligence*, Springer (2009).

[5] Satoh, K. et al.: PROLEG: the Presupposed Ultimate Fact Theory by PROLOG Technology. *Information Network Law Review*, Volume 10, 54-89 (2011).

[6] Satoh, K. et al.: On Generality of PROLEG Knowledge Representation. In *Proc. of JURISIN2012*, 115-128 (2012).