1E4-03

矛盾した法的知識集合からの極小矛盾集合の抽出

Extraction of minimal inconsisntent sets from legal knowledge

萩原 信吾[*1] 東条 敏[*1]
Shingo Hagiwara Satoshi Tojo

[*1]北陸先端科学技術大学院大学情報科学科
Japan Advanced Institute of Science and Technology

In logical representation of legal knowledge, which consists of inference rules, facts, and arbitrary interpretations may include inconsistent propositions. In this paper, we propose an efficient algorithm to extract minimal inconsistent sets out of a legal knowledge, that could help reasoning of a legal agent. In order to reduce the computational complexity, we divide the algorithm into the following two steps. Firstly, we divide the knowledge into multiple cliques, each of which consists of subformulae of a certain formula, to decrease the size. Secondly, we extract arguments, that is a chain of inferences, from a clique. Thereafter, we merge several arguments in conflict with each other. Hence, we acquire the minimal inconsistent sets.

1. はじめに

法的推論で用いられる知識集合は，ルール，ファクトなどの様々な知識が論理式で表現されている．また，論理式への解釈の時点でも多くの揺れを含む可能性がある．そのため，その知識集合は無矛盾性が保証できず，通常の推論ではない何らかの方法が必要となる．なぜならば，古典主義論理において矛盾した知識集合からは任意の論理式が演繹可能であるので推論の結果が意味を持たないからである．

このような矛盾した知識集合から推論する事に関して，[Tahara 04, Roos 92] などの様な研究がある．両者とも，どのような知識が利用可能かを，極大無矛盾集合を用いて検出する方法をとっている．田原らの方法は，すでに極大無矛盾集合が取り出されることが前提であり，その詳細な手順は問われておらず，そのまま矛盾した知識から推論するシステムに組み込むことは出来ない．また Roos らの方法は，信頼関係順序（Reliability Relation）が定義されていることが前提であり，一般的な知識集合に対しそのまま適応できるものではない．これ以外には，極大無矛盾集合を考える一般的な問題として，MAX-SAT 問題が考えられる．

その MAX-SAT 問題の研究としては，[Lewin 02, Egly 01] のようなものがある．しかし，MAX-SAT 問題は NP-complete であることが分かっており，計算量の問題からこの方法はそのまま適応できるわけではない．このような MAX-SAT 問題に関しては NP-complete であることから，近似解のアルゴリズムが研究される．しかし，法的推論のような問題を取り扱う場合には，その解が近似解である方法を一般にとることは得策ではない．演繹結果が論理的に間違っていてはならないからである．以上のことから，一般に矛盾した知識集合からの推論において使用される極大無矛盾集合を用いて推論を行うことは，具体的なシステムを考えた場合，現実的ではないと考えられる．

そこで，本研究では，2CNF[*1]から極小矛盾集合を抽出することとした．

まず，取り扱う問題に関して考える．通常，知識集合は，一般的な論理式が知識集合の要素として許容されたものであるが，その場合は取り扱う問題が大きすぎることが分かっている．そこで，知識集合は法的推論において用いられるような知識を想定しているので，法的推論に焦点をあて，問題に制約を制約を設けることを考える．これにより，適応範囲を損なうことを最小限にしたまま，問題の複雑さを減らすことが可能である．

まず，法的推論において重要なものは議論（Argument）である．議論とは，ある結論を演繹可能な知識の集合である支持（Support）とその結論（Conclusion）の二つ組で表現される．また，議論において重要なものはルールとファクトである．これらのことから取り扱う知識集合には，単純なルールとファクトだけが含まれるという制約をかすことができる．よって，本論文で取り扱う知識集合は，2CNF で表現できるものとする．

そして，その知識集合から矛盾点を抽出するために，2 段階の手順を経る．まず，大きな集合を取り扱うには，それだけで手間がかかる．そこで，知識集合をいくつかの部分集合であるクリーク（Clique）に分割する．そうすることで，取り扱う集合を小さくできる．なお，法的推論で扱う知識は元々ある程度構造化されているので，複数個の Clique に分割できることが十分期待できる．

次に，それらの分割したクリークから矛盾を検出する．矛盾を引き起こしている命題は，それぞれそれを結論とするような支持が存在する．つまり，その矛盾を起こしている命題のそれぞれの議論をあわせたものが極小矛盾集合と考えることが出来る．その極小矛盾集合の定義は以下に従うとする．

定義 1 極小矛盾集合：Σ は論理式の集合．Γ は Σ の極小矛盾集合である．そのとき，$\Gamma \vdash \bot$，且つ，$\forall\phi[\phi \in \Gamma, \Gamma - \{\phi\} \nvdash \bot]$ が成り立つ．

このような，2 段階の手順を踏むことで，具体的な極小矛盾集合の抽出方法を本論文では提案する．従って，本論文は次のような構成を取る．2 章ではクリークの具体的な構築方法をのべ，その際にクリークという部分問題から全体の極小矛盾集合を抽出可能である事をしめす．3 章では，クリークから具体的に極小矛盾集合を抽出する手続きを述べる．そして最後の 4 章ではまとめと，今後の課題について述べる．

2. 法的知識のクリークへの分割

極小矛盾集合を取り出すことを前提に，全体の知識集合を部分問題に分割することを考える．本論文では知識全体を，取り扱うサイズを軽減するために，部分集合に分割する．その部

連絡先: 北陸先端科学技術大学院大学情報科学科，石川県能美市旭台 1-1，Tel: 0761-51-1111，Fax: 0761-51-1149，s-hagiwa2jaist.ac.jp,tojo@jaist.ac.jp

*1 Two-Conjuctive-Normal-Form

分集合のことを，クリークと呼ぶ．このクリークは，クリークから取り出した極小矛盾集合が全体の知識の極小矛盾集合で無ければならない．従って，クリークにはある矛盾を引き起こすような論理式の集合がある場合，そのすべてが一つのクリークに含まれるという仕様が要求される．よって，クリークを形成する場合，論理式の相互の関連を考慮しなければならない．ある論理式 ϕ と ψ が演繹上関係があるかどうかを判断する材料として候補に挙がるものとしてそれぞれの論理式が共通した部分論理式を含むかどうかという観点がある．ただ，論理式を部分論理式に完全に分解するには手間がかかるため，同様の効力を持ち，且つ簡素な手続きで抽出可能なものとして命題変数の名前がある．これは我々が古典主義論理を想定していることにも依存する．そこで，論理式から命題変数を返すような関数を以下のように定義する．

定義 2 命題変数関数 PV: Σ を論理式の集合と仮定したとき，PV は引数の論理式が含むすべての命題変数を返す関数とする．

従って，この関数は以下の様な振る舞いを示す．

もし，Σ を

$$\Sigma = \{(a \vee (\neg b \rightarrow c)) \wedge d, p \rightarrow q\},$$

と仮定した場合，関数 PV は

$$PV(\Sigma) = \{a, b, c, d, p, q\}.$$

となる．

従って，この関数を用いて，クリークは以下のように定義される．

定義 3 クリーク : Σ を知識の集合とする．このとき，ϕ, ψ は任意の論理式である．もし Δ が Σ のクリークである場合，$\forall\phi\psi[\phi \in \Delta, \psi \in \Sigma,$ かつ，もし $PV(\{\phi\}) \cap PV(\{\psi\}) \neq \emptyset$ が成り立つならば $\psi \in \Delta]$ が成り立つ．

このように定義されたクリークにおいては，先の仕様で求められた，「矛盾の包括性」が成り立つ．これは言い換えると，任意の 2 つのクリークからそれぞれ取り出された任意の部分集合の集合論理和は矛盾しないと言うことである．

定理 1 δ はあるクリークの部分集合で，δ' はその他の任意のクリークにおける部分集合と仮定する．そのとき，

$$\forall\delta\delta'[\delta \cup \delta' \nvdash \bot].$$

が成り立つ．

証明 1 Δ と Δ' はクリークである．また，ξ は論理式，Φ, Ψ は論理式の集合で，Φ と Ψ は $\Phi \subseteq \Delta$ かつ $\Psi \subseteq \Delta'$ を満たす．$\Phi \nvdash \xi$ かつ $\Psi \nvdash \xi$ が成り立つと仮定する．そのとき明らかに $\Phi \cup \Psi \nvdash \xi$ が $PV(\Phi) \cap PV(\Psi) = \emptyset$ によって分かる．従って $\Phi \nvdash \bot$ かつ $\Psi \nvdash \bot$ ならば，$\Phi \cup \Psi \nvdash \bot$ が分かる．□

このことから，このクリークは，矛盾を引き起こす可能性のある部分集合を完全に包括している．そのクリークの具体的な抽出のためのアルゴリズムは以下のようにして提示される．

アルゴリズム 1 クリーク分割

1 Σ を与えられた知識集合とする．また Σ_0 を初期知識集合とし，$\Sigma_0 = \Sigma$ とする．

2 ϕ は Σ_0 に含まれる任意の論理式とする．また Δ_0 を初期クリークとする．そのとき $\Delta_0^0 = \{\phi\}$ である．

3 ψ_i は Σ_j から選ばれた任意の要素の論理式とする．そのとき $0 < i < |\Sigma_j|$ に対して以下の手順を繰り返す．もし $PV(\Delta_i^j) \cap PV(\{\psi_i\}) \neq \emptyset$，であるならば $\Delta_{i+1}^j = \Delta_i^j \cup \{\psi_i\}$ とし，またその他の場合 $\Delta_{i+1}^j = \Delta_i^j$ とする．

5 Δ_k^j を Σ における一つのクリークとする．そこで，$k \geq |\Sigma_j|$ である．

6 $\Sigma_{j+1} = \Sigma_j - \Delta_k^j$ とする．もしこのとき $\Sigma_j = \emptyset$，であるならばこのアルゴリズムは停止する．そうでない場合は，$\psi_0' \in \Sigma_{j+1}$, $\Delta_0^{j+1} = \{\psi_0'\}$ とし，手順 *3* へ移る．

3. 極小矛盾集合

この章では，具体的に与えられたクリークから極小矛盾集合を集出する方法を述べる．まず，使用する知識の形式に標準形を与える．これは，取り扱う問題を，2CNF に制限し，後に知識をグラフとして利用するための準備となる．その後，矛盾点の分類を行い，矛盾から導出される知識に含まれないリテラルの検出の際に利用する．最後に，対立する議論と議論を組み合わせ，極小矛盾集合の具体的な抽出を行う．

3.1 含意標準形

我々は，知識集合に含まれる論理式に対して制限を加える．これは，後に述べるアルゴリズムにおいて扱いやすい形式に限定するとともに，その扱う問題の複雑さを限定するためである．その形式は以下に従う．

定義 4 含意標準形 : Σ を知識集合，ϕ は Σ の要素である論理式，C を任意のクリーク，R をルールまたはファクト，そして p をリテラルと仮定する．このとき，知識集合の標準形式は以下に従う：

$$\Sigma = \bigcup_i \phi_i$$
$$\phi_i = \bigwedge_j R_j^{\phi_i}$$
$$R_j^{\phi_i} = p_k \rightarrow p_l$$

ここで i, j, k, l は添え字となる任意の自然数．特に $\top \rightarrow p_k$ はファクトを表し，そのときの $\top$ は常に $true$ と判定される論理定数である．この形式は $2CNF$[*2] と論理的に等しい．

3.2 グラフへの変換

前節の標準形で表現された知識集合は，その含意の関係をそのまま，グラフに変換可能である．従って，計算機で実装を行うことを考え，命題の含意関係をグラフに置き換える．こうすることで，具体的な演繹を行うのではなく，グラフの探索によって同様のことが出来るようになる．そこで，まず，知識集合をグラフに変換する関数を以下のように定義する．

定義 5 グラフ変換関数 : Σ を含意標準形によって表現された知識集合と仮定する．また，$G = \langle V, E\rangle$ はグラフである．そのとき，$CG(\Sigma) = G$ となる関数 CG を定義する．そこで，$p \rightarrow q \in \Sigma$ は $p, q \in V$ と $(p, q) \in E$ として変換されるものとする．

*2 Two Conjunctive Form[Lewin 02]

またグラフ操作を知識集合における推論として扱うために，以下に推論連鎖を定義する．

定義 6 推論連鎖: Σ を知識集合，$G = \langle V, E\rangle = CG(\Sigma)$ をそのグラフ，v, v' をグラフのノード，そして i, j を任意の自然数とする．このとき $CI^i_{v,v'} \subseteq E$ はグラフ G における v から v' へのある経路を表すとする．したがって，$CI^i_{v,v'}$ を知識集合 Σ における v から v' への推論連鎖と呼ぶ．なお，$i \neq j$ であるとき，$CI^i_{v,v'} \neq CI^j_{v,v'}$ が成り立つ．

以後は，これらの関数を用い，知識をグラフとして利用する．

3.3 矛盾による証明の分類

論理式は，論理的演繹の仮定で起こる矛盾を根拠にして，知識集合に含まれていない新たなリテラルが証明される場合がある．例えば

$$\Delta = \{a \rightarrow \neg b, a \rightarrow b\}$$

この論理式の集合は $(a \rightarrow \neg b) \wedge (a \rightarrow b)$ と等しいことから，演繹の結果 $\Delta \vdash \neg a$ が成り立つ．これは，論理的帰結集合（consequence set）$Cn(\Sigma)$ に $\neg a$ が含まれていることを意味する．しかしながら，一般にこの $Cn(\Sigma)$ を考えるのは証明器を用いる必要があるなど，非常に計算手間がかかり，現実的な議論となりにくい．そこで，本論文では，以下のように，現在使用している形式の特性を利用し，矛盾の起こるパターンを 3 つに分類することで，証明器を用いずに $Cn(\Sigma)$ を考慮する．その矛盾を根拠とした証明の分類は以下に従う．

定義 7 帰結矛盾：p, q をリテラル，また $CG(\Sigma) = \langle V, E\rangle$ であると仮定する．このとき Σ が帰結矛盾 を持っている場合，$CI(p, q), CI(p, \neg q) \subseteq E$ が成り立つ．

定義 8 前提矛盾： p, q をリテラル，また $CG(\Sigma) = \langle V, E\rangle$ であると仮定する．このとき Σ が*premise inconsistent* を持っている場合， $CI(p, q), CI(\neg p, q) \subseteq E$ が成り立つ．

定義 9 経路矛盾 p, q をリテラル，また $CG(\Sigma) = \langle V, E\rangle$ であると仮定する．このとき Σ が経路矛盾 を持っている場合，$CI(p, q), CI(q, \neg p) \subseteq E$ が成り立つ．

このような，3 パターンの矛盾の出現は，グラフの探索によって検出される．これは，間接的に証明を行っていることと等しく，含意標準系で表現された知識集合に含まれていない新たなリテラルを $Cn(\Sigma)$ から見つけ出すのに特化した方法である[*3]．

3.4 極小矛盾集合の抽出アルゴリズム

本アルゴリズムはグラフの部分集合として極小矛盾集合を抽出する．このアルゴリズムは大きく分けて 3 つのステップに分けることができる．まず 1 ステップは，各ノードの上方下方経路を抽出する．また，その上方下方経路は以下のようにして定義される．

定義 10 上方下方経路 v の上方経路とは v から出発し，有向グラフを末端までたどった経路のことである．また v の下方経路とは，v から有向グラフを逆向きに可能な限りたどった経路のことである．

次にステップ 2 では，リテラルの証明の経路が存在するかどうかが探索される．まず，リテラルに証明が存在するかどうかは以下の 3 通りが考えられる．それは，(1)$\top$ からの経路が存在する．(2) 下方経路に証明が存在するリテラルが含まれる．(3) 上方下方経路を 3 種類の矛盾による証明の定義 7–8 によって分類した場合，適合するものがある．などによってこのステップ 2 ではリテラルが証明可能かどうかをグラフによって探索を行う．

ステップ 3 ではそうして得られたリテラルと，その論証となる経路を用いて極小矛盾集合を構成する．これは，ある論証経路を持つ v と $\neg v$ が存在した場合，それぞれの論証は知識集合全体において極小に対立する．したがって，

$$\{Arg_{i1}, Arg_{i2}, Arg_{i3}, \cdots\} \rightarrow v$$

$$\{Arg_{j1}, Arg_{j2}, Arg_{j3}, \cdots\} \rightarrow \neg v,$$

のような論証集合が得られていた場合，これらそれぞれの集合論理和をとった

$$\{\{Arg_{i1}, Arg_{j1}\}, \{Arg_{i2}, Arg_{j2}\}, \{Arg_{i3}, Arg_{j3}\}, \cdots\}.$$

がそれぞれ極小の矛盾を構成することになる．なお，関数 LF を論理式に含まれるリテラルを返す関数としたとき，極小矛盾集合のために組み合わせる $\phi, \neg\phi$ の二つの支持議論 Arg,Arg' について，これらは $\psi \in$ Arg,$\psi' \in$ Arg', について矛盾しないとする．このためには，ψ, ψ' の支持議論が Arg$\cup$ Arg' の部分集合であってはならないことが求められる．たとえば，$\Delta = \{a, d, a \rightarrow b, b \rightarrow \neg c, d \rightarrow \neg b, \neg b \rightarrow c\}$ という知識集合を考えた場合，$c, \neg c$ に対する支持議論はそれぞれ $\{d, d \rightarrow \neg b, \neg b \rightarrow c\}$, $\{a, a \rightarrow b, b \rightarrow \neg c\}$ であるが，これを組み合わせた極小矛盾集合は $b, \neg b$ に対する支持議論を組み合わせた極小矛盾集合を完全に含む．したがって，極小矛盾集合とならないのである．以下に具体的な極小矛盾集合を抽出するアルゴリズムを示す．

3.5 極小矛盾集合の抽出アルゴリズム

まず，アルゴリズム内において使用されている変数と関数の定義を以下に行う．

- Σ を知識集合としたとき，Σ は論理式の集合である．
- Δ は以下の定義にしたがう．$\bigcup_{\phi_i \in LF(\Sigma)} \{\phi_i, \neg\phi_i\}$
- $Facts(\Sigma)$ は Σ におけるファクトの集合である．
- $\texttt{Graph} = \langle V, E\rangle = CG(\Sigma)$.
- 関数 $All\_Lower\_Paths(\phi, Graph)$ は ϕ の下方経路集合を返す[*4]そして， $All\_Upper\_Paths(\phi, Graph)$ も同様に ϕ の上方経路集合を返す．
- $Path(v, v', Graph)$ は v から v' への経路集合を返す．
- $NF(E)$ はエッジの集合に含まれるノードの集合を返す E.
- 関数 LF は引数の論理式に含まれるリテラルを返す関数である．

さらに，本論文で定義したその他の関数を用いる．実際のアルゴリズムは以下のようになる．なお，重要な部分についてはプログラミング的に表記を載せる．

---

*3 ただし，L をリテラル，$\neg L$ を L の否定としたとき，$\neg\neg L \equiv L$ で有るとする．

*4 たとえば，$Graph = \langle\{a, b, c\}, \{(b, a), (c, a)\}\rangle$ であるとき，$All\_Lower\_Paths(a, Graph) = \{\{(b, a)\}, \{c, a\}\}$ である．

```
//Check the proof which is given
//by inconsistency
for φ ∈ Δ do
  if (Proof_Set¬φ := Proof_Set¬φ ∪
      CCI(Upper_Pathφ) ≠ ∅ then
    Proved := Proved ∪{¬φ};
  end if;
  if (Proof_Setφ := Proof_Setφ ∪
      CPI(Lower_Pathφ) ≠ ∅ then
    Proved := Proved ∪{φ};
  end if;
  if (Proof_Set¬φ := Proof_Set¬φ ∪
      CAI(Upper_Pathφ) ≠ ∅ then
    Proved := Proved ∪{¬φ};
  end if;
end for;
```

図 1: アルゴリズム 2–3

アルゴリズム 2 極小矛盾集合の抽出

1. $\phi \in LF(\Sigma)$ における全ての ϕ に対し上位下位経路の集合を得る．これは後の演算に使用されるデータである．これら ϕ に対する上位下位経路を $Lower\_Paths_\phi$, $Upper\_Paths_\phi$ と表す．

2. Σ に存在するファクトから到達可能なリテラルを証明可能なリテラルとして証明可能リテラル集合 *Proof_Set* に含める．またそのときの到達経路はおのおのの支持議論となる．

3. *1* より得られたパスを定義 *7–8* によって検査し，ファクト由来ではではなく証明可能なリテラルを抽出し *Proof_Set* に含める．また，このときのルールの集合はそれぞれの支持議論となる．

4. 証明可能なリテラルのうち，その否定も *Proof_Set* に含まれているかどうかを調べる．*(*つまり，もし $\phi \in Proof\_Set$，ならば $\neg\phi \in Proof\_Set$ で有るかど組み合わせ極小矛盾集合を作る．

謝辞

本研究成果は文部科学省 21 世紀 COE プログラムによるものである．

4. まとめ

本研究では，我々は制限した形式である $2CNF$(Two Conjuctive Normal Form) をもちいて形式化した知識集合について，極小矛盾集合の具体的な抽出方法を提案した．

我々はまず，知識集合を有向グラフとみなした．つまり，極小矛盾集合はリテラルをノードとするグラフのパスをたどることによって抽出される．これによって，一般的な証明器を用いることなしに，論理的帰結集合を考慮した極小矛盾集合の抽出が可能になる．

本アルゴリズムは我々の有る仮定に基づいている．それは，我々が用いる法的知識は本来高階の構造を持っており，依存関係のある知識ごとにクリークと呼ばれる知識の部分集合へ分割

```
//Construct minimal inconsistent set
for φ ∈ LF(Σ) do
  for p ∈ Argφ do
    for q ∈ Arg¬φ do
      if ∀ψ, ¬ψ ∈ LF( p ∪ q )
        and PV(ψ) ≠ PV(φ)
        and Argψ ⊆p ∪ q
        and Arg¬ψ ⊆ p ∪ q
      then
        else
          MIS:=MIS ∪ {p ∪ q};
        end if;
      end for;
    end for;
  end for;
```

図 2: アルゴリズム 2–4

可能であるという仮定である．これは，集合のサイズに手続きの計算量が指数的に増えることへの対処である．また，この極小矛盾集合の抽出において重要なことは，この極小矛盾集合は法的推論を行うような人間にとって非常に有用であることである．なぜならば，この極小矛盾集合は知識のなかで矛盾を引き起こしている原因を提供することが可能であり，さらにどのような議論が対立しているのかも提示できるからである．

現在，我々のアルゴリズムは $2CNF$ に限って動作が保証される．この形式化を利用した理由は，法的知識を論理式の知識集合として表現した場合，その集合のサイズは一般的に巨大なものになる．そうした知識集合が完全な論理体系から形式化されたものであるとした場合，そこから極小矛盾集合を抽出するには大変大きな計算量がかかるからである．したがって，演繹時に計算量の観点からより効果的で，また知識としての記述能力の高い法的知識の形式化手法と，その演繹手法が必要であると考える．

参考文献

[Egly 01] Egly, U. and Tompits, H.: Proof-complexity results for nonmonotonic reasoning, *ACM Trans. Comput. Logic*, Vol. 2, No. 3, pp. 340–387 (2001)

[Lewin 02] Lewin, M., Livnat, D., and Zwick, U.: Improved Rounding Techniques for the MAX 2-SAT and MAX DICUT Problems, in *Proceedings of the 9th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pp. 67–82, Springer-Verlag (2002)

[Roos 92] Roos, N.: A logic for reasoning with inconsistent knowedge, *Artificial Intelligence*, Vol. 57, pp. 69–103 (1992)

[Tahara 04] Tahara, I. and Nobesawa, S.: Reasoning from Inconsistent Knowledge Base, *The IEICE Transactions on information and systems, PT.1*, Vol. J87-D-I, No. 10, pp. 931–938 (2004)