目次

図目次

表目次

第1章　はじめに

1.1　研究背景

議論理論とは，文字通り人間の議論をモデルとした理論である．人間が議論するときでは，自分が持つ知識から適切な意見を提示しながら，相手の主張を反駁する．このような論証と攻撃を形式化した形が，議論理論の基本形である．

1995 年 Dung が抽象議論フレームワークを提案[4]して以来，議論理論は急速的に発展しつつある．人の選好または物事の優先度を考慮する議論理論である「プリファレンス付き議論フレームワーク」（PAF）[6]が提案される．これを用いて，法廷の論争では法令間の優先度が考慮しながら結論を導くことが可能になる．また，「価値」と「聴衆（audience）」の概念を導入し，「価値に基づく議論フレームワーク」（VAF）[7]が提案された．これにより，同じ知識を持つ人でも，それぞれの価値観が異なり，違う立場に立っていると，論争で得た結論が異なるということが証明された．また，抽象議論フレームワークが持つ論証は抽象的な概念であり，構造がないため，構造ある論証を取り扱うことが可能な「仮説に基づく議論フレームワーク」（ABA）[8]が提案された．この議論フレームワークは演繹システムを持ち，推論規則や仮説といった古典ロジックにおける論理式の取り扱いが可能になった．さらに，それぞれ異なる知識を持ち，そこから異なる結論を提出しながら，互いに対立関係を持つような議論モデルがあり，それに基づき，新たな議論理論が提案された[3]．この議論理論において，議論間に対立関係を持つのは，VAF のように互い価値観が異なることではなく，各自が持つ知識間に矛盾かあるからである．よって，矛盾ある知識集合が議論理論を用いて解決することが可能であることがわかる．

また，推理に必要な情報が全部文書内に書かれてあるが，推理小説の一つ重要な特徴である．しかしながら，情報が全部示されているのに，簡単に真実にたどり着くことができない．それは，推理小説で示したすべての情報と推理に必要な常識を一つの知識集合とすると，この知識集合内に矛盾があり，「犯人が誰か」という唯一な結論を出せないからである．つまり，推理小説における推理をすることは，推理小説で作られた矛盾ある知識集合から結論を出すのと同じことである．

1.2　研究目的

矛盾ある知識集合の取り扱いは一般的に，無矛盾性に基づき極大無矛盾知識集合を求めるという方法[1]または知識の根拠や根拠の頑強度といった概念により，根拠が弱い知識を捨てるという方法が採用されるが，本研究では議論理論を用いて，推理小説における矛盾ある知識集合を取り扱う．これにより，「犯人が誰か」という結論を出す．

本研究では，推理小説から矛盾ある知識集合を求める方法を提案し，そこから議論を得る．それを用いて知識集合の矛盾の発見及び排除を行うプログラムを実装することにより，推理小説中の犯人を探ることである．

また，推理小説からの矛盾ある知識集合は一般の矛盾ある知識集合と比べると，特殊なところがある．それは，推理小説では地の文で書かれたことは一般的真実であるため，これらで変換した知識も必ず真であることである．本研究では，このような特殊な議論の研究により，条件付きの議論理論研究に貢献できると考える．

1.3　本稿の構成

本章では研究の背景および目的を説明した．第 2 章では本研究の理論基礎である議論理論を紹介する．第 3 章では本研究におけるシステムの構造述べ，アルゴリズムとシステムの実装についても説明し，また，研究の結果を述べ，システムを評価する．第 4 章では本研究のまとめ，および今後の課題について説明する．

第2章　議論理論

本章では，本研究の理論基礎となる議論理論を紹介する．

2.1　Dung の抽象議論フレームワーク

人間が議論を行う場合は，一般的な流れとして，各自が根拠ある意見を提示しながら，議論の相手の意見を攻撃する．ここで紹介する Dung の抽象議論フレームワークは，人間の議論の流れを形式化した枠組の一つである．

2.1.1　基本定義

定義 2.1.1（抽象議論フレームワーク）[5]

議論フレームワーク（argumentation framework: AF）は，抽象的な議論（abstract argument）の集合 A と A 上の関係 R（すなわち，R ⊆ A×A）の対（A, R）として定義される．R は攻撃関係と称され，a R b（すなわち（a, b）∈ R）である時，「論証 a は論証 b を攻撃する」という．抽象的な論証は特別な構造をもたず，単に他の論証を攻撃するか，他の論証から攻撃されるものである．しばしば議論フレームワーク AF＝（A, R）は，A の任意の論証 a を節点とし，a R b を節点 a から節点 b への有向辺とする有向グラフで表現される．

Dung の論文[4]より下の例を挙げる．

例 2.1 人間 I と A が議論している．

I：「私の政府はあなたの政府と交渉できない．それは，あなたの政府は私の政府を認めていないから」

A：「あなたの政府も私の政府を認めていない」

I：「しかし，あなたの政府はテロリスト政府だ」

以上三つの発言を i_1，a，i_2 とすると，a と i_1 は互いに反駁し，また，i_2 は a を反駁する。

この I と A の議論を議論フレームワークで表現すると，A＝{i_1, a, i_2}，R＝{(i_1, a)，(a,

i_1), (i_2, a)}となる. また, 図 2.1 が AF のグラフ表示である.

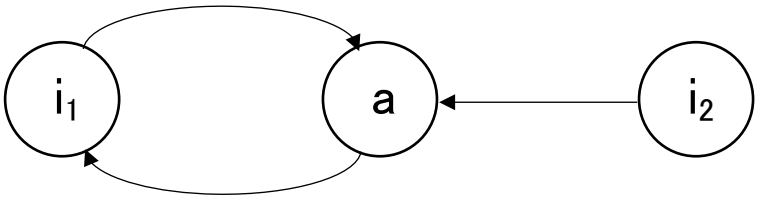


図 2.1 例 2.1 の AF グラフ

定義 2.1.2[5]

論証集合 S⊆A と論証 a ∈ A について, x R a(すなわち, (x, a) ∈ R)なる論証 x ∈ S が存在するとき,「S は a を攻撃する」といい, attacks(S, a)で表す.

例えば, 例 2.1 における議論では, (a, i_1) ∈ R であるので, S＝{a}とすると, S は i_1 を攻撃し, attacks(S, i_1)で表す.

定義 2.1.3(無衝突(conflict-free))[5]

論証集合 S⊆A において, 互い攻撃するような論証が S に存在しなければ,「S は AF において無衝突(conflict-free)である」という. S が無衝突であることを cf(S)で表すと, 無衝突は次式で定義される.

S⊆A について, cf(S) ⇔ ∄a,b ∈ S. a R b

AF のすべての無衝突集合からなる集合を次の CF(AF)で表す.

CF(AF)＝{S | cf(S)かつ(S⊆A)}

論証集合 S が無衝突のとき, S に属する論証すべては互い攻撃や反論しない. ここで, 論証集合 S の要素を知識として考えると, 論証集合の無衝突と知識集合の無矛盾は近い表現であることが考えられる.

例 2.2 例 2.1 の議論フレームワーク AF において, 以下の無衝突な論証集合を持つ.

CF(AF)＝{{}, {i_1}, {a}, {i_2}, {i_1, i_2}}

2.1.2　論証の性質

定義 2.1.4（受理可能（acceptable））[5]

論証集合 S⊆A について，任意の論証 b ∈ A が a を攻撃する（すなわち，b R a）ならば，必ず S はbを攻撃する（すなわち，bを攻撃する論証cが S に存在する）とき，「論証 a ∈ A は論証集合 S⊆A に関する受理可能（acceptable）である」という．「a は S に関して受理可能である」は「S は a を防御する（defends）」ともいい，S defends a，あるいは，defends（S，a）で表す．この受理可能性は次式で定義される．

$$\text{defends}(S, a) \Leftrightarrow \forall b \in A.(b\ R\ a \rightarrow \exists c \in S.c\ R\ b)$$

あるいは，

$$\text{defends}(S, a) \Leftrightarrow \forall b \in A.(b\ R\ a \rightarrow \text{attacks}(S, b))$$

受理可能性は図で表すと以下になる．

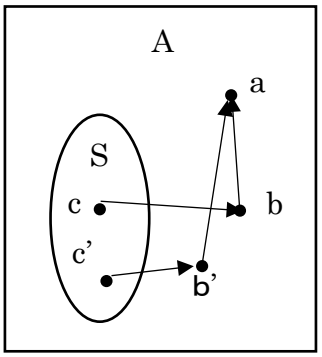


図 2.2　受理可能性

ここでは，方形が論証集合 A を，円形が論証集合 S を表す．点 a，b，c，b'，c' がそれぞれ一つの論証を表す．矢印は攻撃関係を表す．そうすると，論証 a が S に関して受理可能である時，図 2.2 で表示するように，A 中の論証 b，b' が論証 a を攻撃すると，論証集合 S 内には必ず論証 b ，b' を攻撃するような論証cまたは c' が存在する。つまり，S は議論 a の味方の議論集合と見なすことができる．

例 2.3　例 2.1.の議論フレームワーク AF において，以下のものは受理可能である．

$\{\}$ defends i_2，$\{i_1\}$ defends i_1，$\{i_1\}$ defends i_2，$\{i_2\}$ defends i_1，
$\{i_2\}$ defends i_2，$\{i_1, i_2\}$ defends i_1，$\{i_1, i_2\}$ defends i_2

定義 2.1.5（許容可能（admissible））[5]

論証集合 S⊆A が無衝突，かつ S に属する任意の論証 a ∈ S を S が防御する（すなわち，受理可能である）ならば，そしてそのときに限り，「S は許容可能（admissible）」とい

い，そのような S を許容可能集合という．S⊆A の許容可能性は次式で表される．

admissible(S) ⇔ cf(S)かつ∀a (a ∈ A → defends(S, a))

許容可能性は図で表すと以下になる．

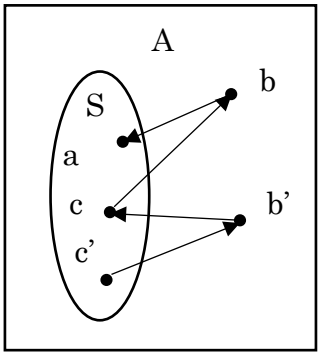


図 2.3 許容可能性

ここでは図 2.2 と同様に，方形が論証集合 A を，円形が無衝突な論証集合 S を表す．点 a, b, c, b', c'がそれぞれ一つの論証を表す．矢印は攻撃関係を表す．そうすると，論証集合 S が許容可能である時，図 2.3 で表示するように，論証集合 S 中の論証 a が論証 b に攻撃されると， S 内には論証 b を攻撃するような論証c存在する。また，論証 c が論証b'に攻撃されると， S内にはさらに論証b'を攻撃するような論証c'存在する．つまり，許容可能な論証集合 S では，S に属する要素である仲間を集合 S 外の論証の攻撃から守ることができる．

例 2.4 例 2.3 の各無衝突な論証集合について，許容可能性を調べる．

(1) S={a}は無衝突である．a ∈ S に対して，i_1 R a と i_2 R a の攻撃がある．i_1 R a について a R i_1 なるヨa ∈ R が存在するが，i_2 R a に対して i_2 を攻撃する論証が S に存在しない．よって S は a を防御(defends)しないので，S={a}は許容可能ではない．

(2) S={ i_1, i_2}は無衝突である．例 2.3 により，{i_1, i_2} defends i_1 かつ{i_1, i_2} defends i_2 であるので，S={ i_1, i_2}は許容可能である．

(3) 同様の計算により，{}，{i_1}，{i_2}も許容可能集合として得られる．

2.1.3 議論意味論

Dung の標準的な議論意味論は完全意味論，選好意味論，安定意味論，基礎意味論であり，それぞれ独自の拡張を持ち，以下により定義する．ただし，議論フレームワーク AF

＝(A, R)をする。

定義 2.1.6(完全拡張)
　論証集合 E⊆A は完全拡張(complete extension)である.
⇔E は許容可能かつ E に関して受理可能な論証はすべて E の要素である.

定義 2.1.7(選好拡張)
　論証集合 E⊆A は選好拡張(preferred extension)である.
⇔E は包含関係⊆に関して極大の許容可能集合である.

定義 2.1.8(基礎拡張)
　論証集合 E⊆A は基礎拡張(grounded extension)である.
⇔E は包含関係⊆に関して極小の完全拡張である.

定義 2.1.9(安定拡張)
　論証集合 E⊆A は安定拡張(stable extension)である.
⇔E は無衝突, かつ A＼E に属する任意の論証を攻撃する.
　すなわち, E ∈ CF(AF)かつ∀a ∈ A(a ∉ E → attacks(E, a)).

　以上で定義した拡張間の関係は図で表現すると図 2.4 になる. ここでは, 許容可能集合を S で表す。完全拡張, 選好拡張, 基礎拡張, 安定拡張がそれぞれ集合 co, pr, gr, st で表す.

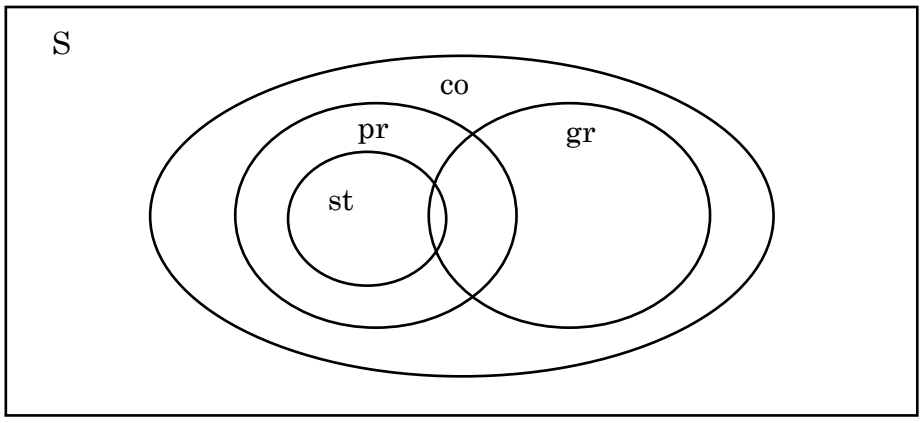


図 2.4　各拡張の関係

　図で表示するように, すべての拡張は許容可能集合の部分集合である. また, 選好拡張, 基礎拡張, 安定拡張は完全拡張の部分集合であり, 安定拡張は選好拡張の部分集

合である. 基礎拡張と安定拡張は前のような包含関係がなく, また, $\mathrm{pr} \cap \mathrm{gr} = \varnothing$ という状況も存在する。

2.2 仮説に基づく議論フレームワーク

仮説に基づく議論フレームワーク(ABA)[8]は Dung の抽象議論フレームワークから発展した議論理論である. ABA は抽象議論フレームワークと同様に非単調推論をモデルかできる一般化された議論理論であり,「仮説」という新たな概念を導入することにより,「知識」ということを表現できるようになった.

2.2.1 基本定義

定義 2.2.1(仮説に基づく議論フレームワーク)[5]

仮説に基づく議論フレームワーク(assumption-based argumentation framework:ABA フレームワーク)は次の四つ組で定義される.

$$\langle L, R, A, {}^{-} \rangle$$

ここで,

(1) <L, R>は演繹システムである. ただし, L は可算個の文からなる論理的言語であり, R は可算(無限)個の推論規則(ルール)の集合である. ここでは, L を次の形で表現する,

$$L = \{a_0, a_1, \cdots, a_n\};$$

また, 推論規則は次の形で表現する,

$$b_0 \leftarrow b_1, \cdots, b_m \quad (m \geq 0)$$

ここでは, a_i または b_j $(0 \leqq i \leqq n,\ 0 \leqq j \leqq n)$は論理的言語を表し, 意味を持つ。また, a_i と b_j が次の式を満足する.

$$\forall j \in (0, n) \rightarrow \exists i \in (0, j)(a_i = b_j)$$

すなわち, 推論規則で使用する言語はすべて L の要素である.

またここでは, ルール r について, b_1 を r の頭部(head)といい, $b_1, \cdots, b_m$ を r の本体(body)という.

(2) $A \subseteq L$(ただし, $A \neq \varnothing$)は, 仮説(assumption)の集合である. また, ここでの ABA フレームワークはフラット(flat)であるため, 仮説 $\alpha \in A$ はルールの頭部に出現しないものとする.

(3) ${}^{-}$は A から L への全関数である. 仮説 $\alpha \in A$ に対して, $\bar{\alpha}$は, "αの相反"(contrary of α)と称される.

例 2.5 以下の例を ABA フレームワークで表現する.

「北回帰線の北にいれば七月は夏である．七月である．正午のとき太陽が北にあると北回帰線の北にいない」

ABA フレームワーク<L, R, A, ¯>では以下の形で表現できる．

$$L=\{a, b, c, d, e, f, g, h\}$$

$$R=\{d \leftarrow e, a,\ \ e \leftarrow,\ \ f \leftarrow b, c\}$$

$$A=\{a, b, c\}$$

$$\bar{a}=f,\ \bar{b}=g,\ \bar{c}=h,$$

ここで，a は「北回帰線の北にいる」，b は「正午である」，c は「太陽が北にある」，d は「夏である」，e は「七月である」，f は「北回帰線の北にいない」，g は「正午ではない」，h は「太陽が北にない」を表す．また，例の言葉で含まれる知識は R で表現し，各言語（a, b…）の関係性を表す．

定義 2.2.2（ABA の論証）[5]

ABA フレームワーク<L, R, A, ¯>において，“K にサポートされたクレーム c ∈ L の論証（an argument for the claim c supported by K）”は K ⊢ c で表し，K⊆A と r⊆R から演繹的に c が推論されることを意味する．

例えば，K＝{a}，また，r＝{b ← a}⊆Rとすると，ロジックにおける演繹法ではbが推論され，K ⊢ bで表現する。

例 2.5 の ABA フレームワーク<L, R, A, ¯>は，以下の論証を持つ．

$$\{a\} \vdash d,\ \ \{\} \vdash e,\ \ \{b, c\} \vdash f,\ \ \{a\} \vdash a,\ \ \{b\} \vdash b,\ \ \{c\} \vdash c$$

例の言葉で表現する知識が持っていれば，論証を以下のように表現できる．

（1） {a} ⊢ d：北回帰線の北にいると，夏である．（知識「七月である」（e ←，）と「北回帰線の北にいれば七月は夏である」（d ← e, a）を用いる）

（2） {} ⊢ e：夏である．（知識「七月である」（e ←，）を用いる）

（3） {b, c} ⊢ f：正午である，また太陽が北にあると，北回帰線の北にはいない．（知識「正午のとき太陽が北にあると北回帰線の北にいない」（f ← b, c）を用いる）

（4） {a} ⊢ a：北回帰線の北にいると，北回帰線の北にいる．（知識を使わない）

（5） {b} ⊢ b：正午であると，正午である．（知識を使わない）

（6） {c} ⊢ c：太陽が北にあると，太陽が北にある．（知識を使わない）

2.2.2 論証木と攻撃関係

定義 2.2.3（ABA の論証木）[5]

ABA フレームワーク<L, R, A, $^-$>において, K $\vdash$ c は, 節点が L の要素または τ でラベル付けられた以下の木である. ただし, ここでは, τ は恒真式(tautology)を意味する.

- 根はcでラベル付けられる
- すべての節点 N について,
  (1) N が葉であるならば, N は K に属する仮説, または τ でラベル付けられる.
  (2) N が葉ではなく, そして b_0 が N のラベルならば, b_0 を頭部にもつルール $b_0 \leftarrow b_1, \cdots, b_m$ $(m \geq 0)$が存在し, m＝0 ならば N の子節点は τ でラベル付けられ, $m \geqq 0$ ならば N は $b_i (1 \leq i \leq m)$でラベル付けられた m 個の子節点を持つ.
  (3) K は葉をラベル付けしている仮説からなる集合である.
- このような論証を表す木を論証木(argument tree)と称する.

なお上記定義より, 各仮説 $\alpha \in$ A について, 根が αでラベル付けされた唯一の節点の木が存在し, それらは$\{\alpha\}$でサポートされたクレーム αの論証$\{\alpha\} \vdash \alpha$として表される.

論証木の定義では, L で表現する言語は節点で表す. R で表現する推論規則は親節点と子節点の間に引く線を表す. また, A で表現する仮説と恒真式の τ は葉節点で表す.

例 2.5 の論証木が図 2.5 で表せる.

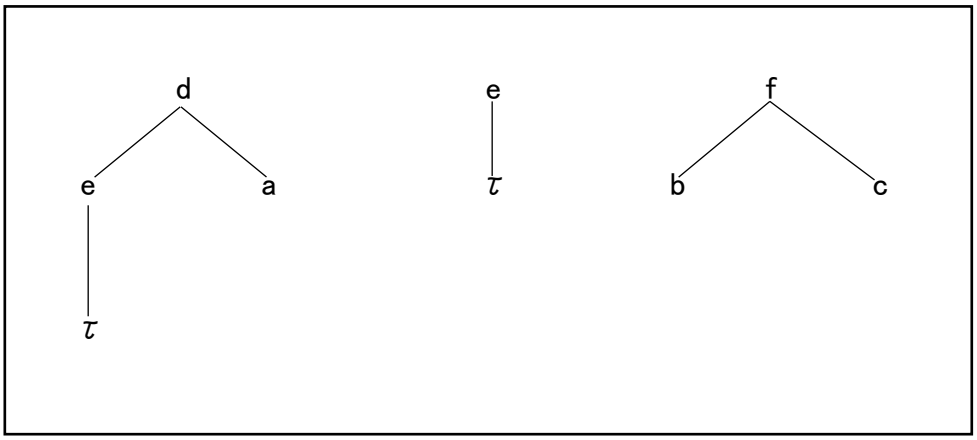


図 2.5 例 2.5 の論証木

ABA における攻撃関係 attacks は, “仮説に反する(contrary of assumption)”の概念に基づいて定義される.

定義 2.2.4(論証間の攻撃(attacks))[5]

- 論証 K $\vdash$ c が仮説 α を攻撃する.
  $\Leftrightarrow$ 論証のクレーム c が仮説 α の相反 $\bar{\alpha}$ である. すなわち, c＝$\bar{\alpha}$.
- 論証 K $\vdash$ c が論証 K’ $\vdash$ c’を攻撃する.
  $\Leftrightarrow$ K $\vdash$ c が K’のある仮説を攻撃する.

すなわち，c＝$\bar{\alpha}$なる仮説 α が K’に存在する.

定義 2.2.5[5]

claim(A)は K ⊢ c なる論証 A のクレーム c を表す. Concs(E)は E に属する論証のクレーム全体の集合. すなわち,

$$Concs(E)=\{c \mid K \vdash c \in E\}$$

を表し，拡張 E の結論集合と称する.

例 2.5 の ABA フレームワーク<L, R, A, $\bar{}$>が持つ論証は以下である.

{a} ⊢ d, {} ⊢ e, {b, c} ⊢ f, {a} ⊢ a, {b} ⊢ b, {c} ⊢ c

{a} ⊢ a のような仮説から自分というクレームを推論する論証以外の論証は，それぞれ図 2.2 のクレームを根とする論証木として表現される. また，仮説の相反を示した($\bar{a}$＝f, $\bar{b}$＝g, $\bar{c}$＝h)である$\bar{}$を用いて，この ABA フレームワークの攻撃関係 attacks を以下であるように得られる.

{b, c} ⊢ f attacks {a} ⊢ d, {b, c} ⊢ f attacks {a} ⊢ a

2.2.3 Dung の抽象議論フレームワークとの関係

仮説に基づく議論フレームワーク(ABA)は Dung の抽象議論フレームワークから発展した議論理論である. ABA フレームワークは抽象議論フレームワーク(AA)に変換でき，また，議論意味論により，完全拡張など抽象議論フレームワークと同様な論証拡張を得られる.

抽象議論フレームワークでは，論証が抽象的な概念であり，構造を持たない. そのため，抽象議論フレームワークにおける論証は一つの物事のみ表現でき，論証間の関係も攻撃するか，攻撃されるかあるいは無関係であるという三つの関係のみ，複雑なモデルには向かない. それに対して，ABA の論証には仮説と推論規則に基づく構造があるため，議論フレームワークをより豊かに表現できる. また，ABA には論証木という形式がある. これにより，提案者と反論者の論争木を構造することが可能であり，そこから議論の勝利者を導くことができる.

2.3 知識に基づく議論理論

人間が行う議論を形式化した枠組は Dung の議論フレームワーク以外他にもある. Besnard[3]または太原[1]では Dung の議論フレームワークと異なる議論理論を採用する.

これらの論文中では単に「議論」または「argument」といった名称を使用しているので，本研究では Dung の議論フレームワークなどと区別するため，この議論理論を「知識に基づく議論理論」という名称とする．それは，この議論理論における議論の支持と結論は知識集合から作られたものであるからである．ただし，ここでの知識集合は命題論理式の集合とする．

2.3.1　基本定義と対立関係

定義 2.3.1（議論）[1]

矛盾した知識集合を Σ とする．命題論理式 ϕ に対して，Σ の部分集合 Φ が

(1) Φ は無矛盾である．

(2) $\Phi \vdash \phi$

を満たす極小集合であるとき，$<\Phi, \phi>$ を Σ からの議論といい，Φ をその議論の支持，ϕ を結論という．

定義により，議論の支持は条件を満たす極小集合であるため，結論は支持である知識集合内のすべての知識を用いて推論できる論理式である．よって，$<\{p, p \rightarrow q\}, q>$ は議論であるが，$<\{p, p \rightarrow q\}, p>$ は議論ではない．それは，$(p \rightarrow q)$ という知識を用いていないからである．

定義 2.3.2(健全性) [1]

知識集合 Σ からの議論 $<\Phi, \phi>$ に対して，Φ の要素がすべて真であるとき，議論 $<\Phi, \phi>$ は健全であるという．

例 2.6　知識集合 $\Sigma = \{p, \neg p, \neg p \rightarrow q, p \rightarrow r\}$ とする．Σ から

$A_1 = <\{p\}, p>$

$A_2 = <\{\neg p\}, \neg p>$

$A_3 = <\{p, p \rightarrow r\}, r>$

$A_4 = <\{\neg p, \neg p \rightarrow q, \}, q>$

…

といった議論が得られる．

また，p, q, r がともに真とすると，A_1 と A_3 が健全であり，A_2 と A_4 が健全ではないという．

定義 2.3.3（議論間の対立関係）[1]

$A_1 = <\Phi_1, \phi_1>$, $A_2 = <\Phi_2, \phi_2>$ を Σ からの議論とする．

(a)（反駁(rebut)）
$\phi_1 = \neg \phi_2$ であるとき，A_1 は A_2 を反駁する(また，A_2 は A_1 を反駁する)という.
(b)（無効化(undercut)）
$\exists \phi \in \Phi_2$ について，$\phi_1 = \neg \phi$ であるとき，A_1 は A_2 を無効化するという.
(c)（不同意(disagreement)）
$\{\phi_1, \phi_2\}$ が矛盾しているとき，A_1 と A_2 は不同意であるという.
(d)（弱無効化(weakly-undercut)）
$\exists \phi \in \Phi_2$ について，議論 $\langle \Phi, \phi \rangle$ で $\{\phi_1, \phi\}$ が矛盾しているとき，A_1 は A_2 を弱無効化するという.

言葉で説明すると，反駁関係は，二つの議論は結論が互いに否定することを意味する. 無効化関係は，議論 A の結論は議論 B の支持の一つを否定することを意味する. 不同意関係は，二つの議論は結論が矛盾することを意味する. 弱無効化関係は，議論 A の支持の部分集合を支持とする議論が，議論 B と不同意であることを意味する.
以上の説明により，議論 A は議論 B を反駁すると，A と B は不同意であることがわかる. また，議論 A は議論 B を無効化すると，A は B を弱無効化することもわかる.

例 2.6 以下の議論を考える[1].

$$A_1 = \langle \{p, p \rightarrow q\}, q \rangle$$
$$A_2 = \langle \{r, r \rightarrow \neg q\}, \neg q \rangle$$
$$A_3 = \langle \{p, \neg q, (p \wedge \neg q) \rightarrow r\}, r \rangle$$
$$A_4 = \langle \{r, r \rightarrow \neg q\}, r \wedge \neg q \rangle$$
$$A_5 = \langle \{r, r \rightarrow \neg q, \neg q \rightarrow p\}, p \rangle$$

この五つの議論の対立関係を分析すると，以下のような結論が得られる.

- A_1 の結論 q と A_2 の結論 $\neg q$ は相反するため，A_1 は A_2 を反駁し，A_2 は A_1 を反駁するという.
- A_1 の結論 q は A_3 の支持中の要素 $\neg q$ と相反するため，A_1 は A_3 を無効化しているという.
- A_1 と A_4 の結論の集合 $\{q, r \wedge \neg q\}$ が矛盾しているため，A_1 と A_4 は不同意であるという.
- A_5 から $\langle \{r, r \rightarrow \neg q\}, \neg q \rangle$ という議論が得られ，この議論は A_1 と不同意であるため，A_1 は A_5 を弱無効化している.

2.3.2 結論の正当化条件

大原[1]では，「知識集合 Σ からの議論 A について，それを無効化する議論が存在しな

ければ，A を反駁する議論，A と不同意な議論，A を弱無効化する議論のいずれも存在しない」ということを証明した．これを基づいて以下の概念を定義する．

定義 2.3.4（有効）[1]
矛盾した知識集合を Σとする．ϕ を結論とする Σ からの議論 A＝〈Φ，ϕ〉について，それを無効化する議論が存在しないか存在してもすべて健全でないとき，A＝〈Φ，ϕ〉は有効であるという．

定義 2.3.4（正当化条件）[1]
矛盾した知識集合を Σ とする．命題論理式 s のもとで ϕ を結論とする Σ からの議論の少なくとも一つが有効であるとき，s を ϕの正当化条件という．

ここで，正当化条件が表現するのは，ある知識集合 Σ からある結論 ϕ を導くために，論理式 s は必ず必要とするということである．
また，知識に基づく議論理論では正当化条件により論争の勝利者を決定することができる[5]．

2.3.3　ABA フレームワークとの関係

本節で説明した議論理論は ABA フレームワークと同様に論証が構造を持ち，推論規則を表すことができる．また，ABA フレームワークにおけるクレームと知識に基づく議論理論における議論は，それを表現する形が相当に似ている．
しかしながら，知識に基づく議論理論では，推論規則と仮説は知識を表す論理式として同様に扱う．また，論理式の表現に制限がない．それに対して，ABA フレームワークでは，推論規則と仮説の扱う方法は異なる．また，ABA フレームワークの仮説は定義により，ルールの頭部に出現しないものとするため，自由に決めることができない．
また，論争の結果を決める方法について，ABA フレームワークでは論証木から導くことに対して，知識に基づく議論理論では正当化条件により導出する．

2.4　本研究で取り扱う議論理論

本研究では，研究対象である推理小説から論理式で表現する知識を取り扱う．よって，論証に構造を持たない抽象議論フレームワークでは知識を表現できない．仮説に基づく議論フレームワークではこのような知識を表現できるが，知識である推論規則と仮説を同じ取り扱うことはできない．知識に基づく議論理論では，知識と結論を議論という形で表現でき，表現が最も適切であるので，本研究では，知識に基づく議論理論を理論基礎と

なる.

また，本研究では知識に基づく議論理論を採用しているが，取り扱う議論は結論が原子論理式であるものに限る. その理由は以下の例で説明する.

例 2.7　議論 A＝<{p, p → ¬q}, p∧¬q >を考える

仮にここで，p は「冬である」，q は「気温が高い」を表す. なら，p → ¬q は「冬であると気温が高くない」を表す. つまり，議論 A の支持は「冬である」と「冬であると気温が高くない」二つある. 一般的な考え方では，この支持からの結論は「気温が高くない」であり，すなわち¬q である. 議論 A の結論は p∧¬q，すなわち「冬である，そして気温が高くない」ということで，二つの知識を表す.

この結論は論理的には正解であるが，人の思考とは異なる. 実際の応用では，一つの議論を支持から多数の結論へ導くより，各議論を支持から一つだけの結論へ導いたほうが，人間の思考に近い. ここの議論 A では，

A_1＝<{p}, p >

A_2＝<{p, p → ¬q}, ¬q >

二つ議論に分けると，A_1 は「冬である」という知識から「冬である」のという結論へ導き，A_2 は「冬である」と「冬であると気温が高くない」二つの知識から「気温が高くない」という結論へ導く. A_1 と A_2 は A の結論をすべて表したうえ，A より人の思考に近いため，本研究における議論は結論が原子論理式であるもののみとする.

第3章　研究方法

本章では，本研究における研究方法を説明する．まず，本研究におけるシステムの構築を紹介する．次に，第2章で説明した議論理論を用いて，ある知識集合から結論が特定なものの議論を計算するアルゴリズムを説明する．そして，本研究の研究対象である推理小説の事件の内容を紹介し，その内容を論理式で表現する知識集合への変換方法を提案し，そこから，推理小説の知識集合を得る．次に，説明したアルゴリズムも使用して，この知識集合から必要な議論を計算し，これにより犯人を見つける．最後，得た解と探偵役が得たものと比較し，システムを評価する．

3.1　システムの構築

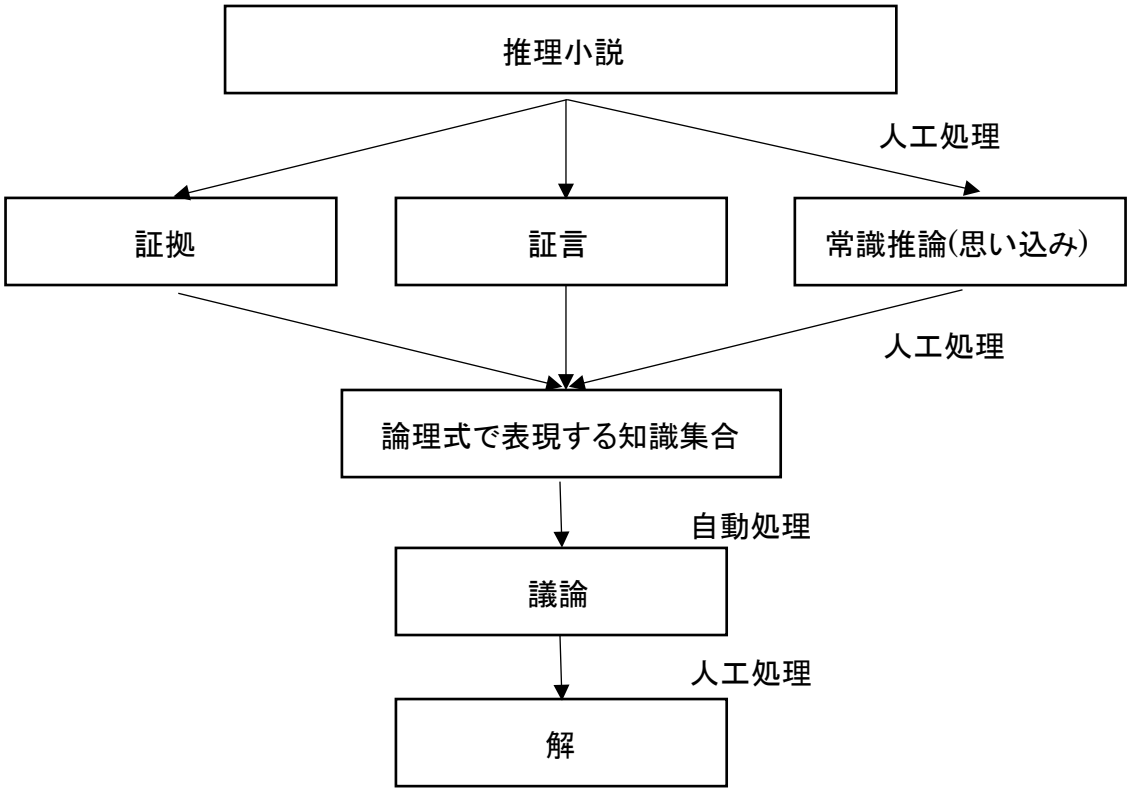


図 3.1　システム構築

本研究におけるシステムは，前のページの図で示す.

システムの流れは，まず，研究対象である推理小説を分析し，推理に必要な知識を「証拠」,「証言」,「常識推論」三つ分ける. 次に，それぞれの知識を論理式に変換し，矛盾ある知識集合を得る. そして，本研究のプログラムを使用して，特定な論理式へ導ける議論，すなわち結論がその論理式の議論を計算する. 次に，得た議論を比較し，システムの解を出力する.

3.2　プログラムのアルゴリズム

本研究のプログラムの基本の形を説明する. プログラムの入力は知識集合 Σとある原子論理式 φで，出力は Σ における結論が φであるすべての議論の支持の集合とする.

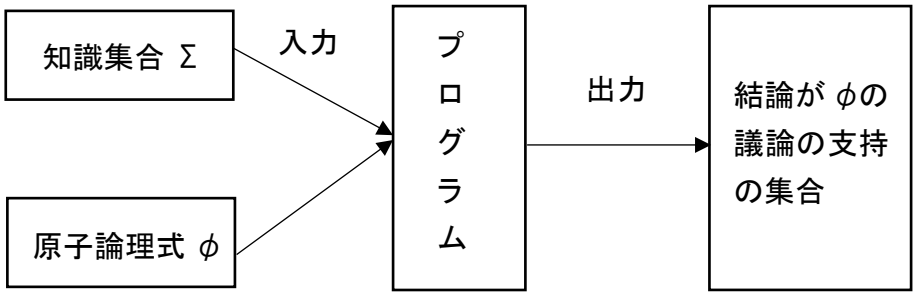


図 3.2　プログラムの入力と出力

本研究におけるプログラムのアルゴリズムの要は，ある木を生成することである. この木は知識集合 Σを部分表現することができる. また，最終的に得た木は，根節点に持つ支持集合がプログラムの出力である.

以上のように，本アルゴリズムの流れは大きく分けると二ステップある. まず，知識集合を用いて木を生成する. 次に，生成した木を使い，根節点に持つ支持集合を計算する. 二ステップとも再帰法を採用する.

3.2.1　木の生成

第二ステップである支持集合の計算をしやすくため，木の節点を「結論節点」と「知識節点」二種類分ける. 結論節点には原子論理式 K＝φを持つ. 知識節点には知識を表す論理式（原子論理式 b_0 または推論規則 b_0 ，b_1，…，b_n→Q）を持つ. また，この二種類の節点は以下の図で表す. 結論節点を円の形式で，知識節点を方形で表現する. また，木の根節点は結論節点であり，原子論理式 K＝φを持つ.

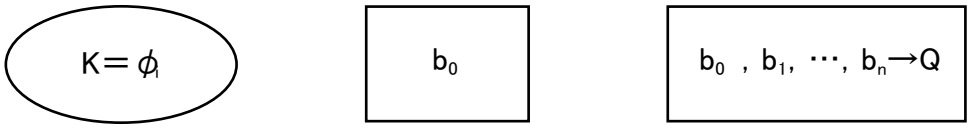


図 3.3 結論節点と知識節点

根節点は最初から与えられた節点であるため，ある節点から子節点を生成する方法と葉節点の判別法がわかれば，それを再帰法により木を生成することができる.

根節点は結論節点であるため，まず，結論節点から子節点の生成方法を説明する. また，本アルゴリズムでは，結論節点の子節点は全部知識節点であり，知識節点の子節点は全部結論節点であるとする.

（1）結論節点から子節点の生成方法

① この結論節点に持つ原子論理式 K＝ϕとする.

② 知識集合 Σから ϕという式または ϕが“→”の右にある式をすべて探る.

③ ②で得た式は，この結論節点の子節点である知識節点に持つ論理式である.

④ ②で得た式がない場合に，この結論節点は葉節点である.

ここでは，②で得た式集合を{ϕ, a_0, a_1, …, a_n→ϕ}とすると，子節点の生成方法を図で表すと以下になる.

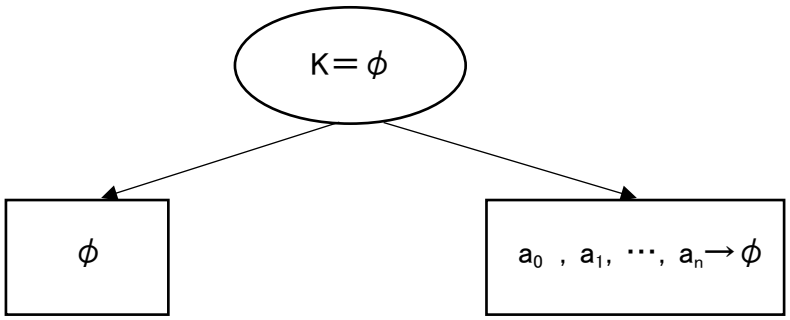


図 3.4 結論節点から子節点の生成方法

また，具体例をあげる.

例 3.1 知識集合 Σ＝{a, b, d, a → d, b, c → d}. 結論節点に持つ原子論理式 K＝d.

知識集合 Σ中に，d または“→ d”のような式を探る. ここでは式 d と式 b, c → d が得られる. よって，図 3.5 のように子節点が得られる.

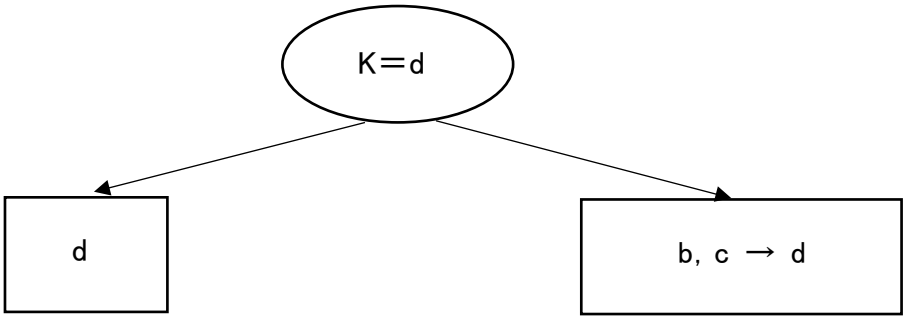


図 3.5　結論節点の子節点生成の具体例

また，例 3.1 の知識集合 Σにおいて，結論節点に持つ原子論理式 K＝cとすれば，Σ中には，c または“→ c”のような式が存在しない．よって，この結論節点が葉節点である．

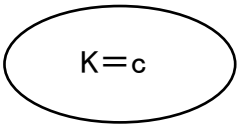


図 3.6　葉節点である結論節点の例

次に，知識節点から子節点の生成方法を説明する．

（2）知識節点から子節点の生成方法

① この知識節点にもつ式は原子論理式であるとき，この節点は葉節点である．

② この知識節点にもつ式は推論式であるとき，“→”左の部分を探り，幾つの原子論理式を得る．

③ ②で得た原子論理式は，この知識節点の子節点である結論節点に持つ原子論理式である．

図で表すと以下になる．ここでは，ϕを原子論理式と見なす．

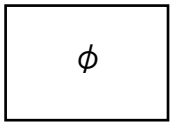
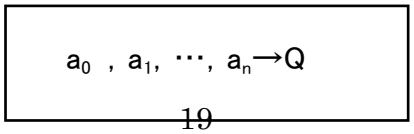


図 3.7　葉節点である知識節点の例



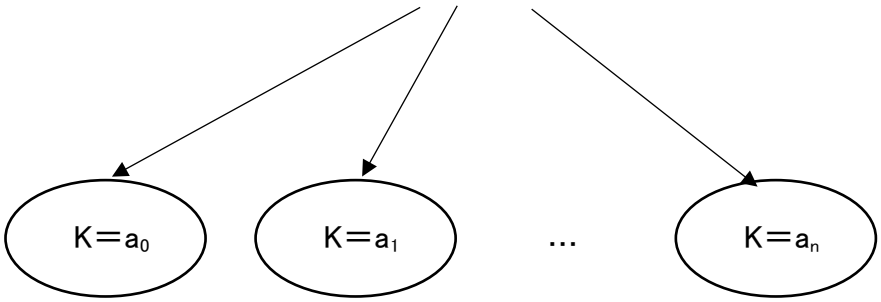


図 3.8　知識節点から子節点の生成方法

（1）と（2）で説明した結論節点または知識節点から子節点の生成方法と葉節点の判別方法により，木を生成できる.

3.2.2　木の生成アルゴリズムの問題点と改善

一般的な知識集合 Σと結論では，以上のアルゴリズムを用いて有限木を生成できるが，もし知識集合 Σが特殊であるとき，本アルゴリズムが停止できず，無限木を生成する恐れがある. 以下の例を考える.

例 3.2　知識集合 Σ＝{p,　p → q,　q → r,　r → s,　s → q }であり，根節点持つ原子論理式は K＝s である.

木の生成過程は次のページの図 3.9 で表現する.

図で示されたように，論理式“s → q”を持つ知識節点から，根節点と同様に原子論理式 K＝s を持つ子節点を生成する. それにより深さが無限になる木ができる. アルゴリズムが停止できない. また，そこから新たなことも得られない.

このような木ができる理由は，知識集合 Σには循環証明があるから. ここでは，

q → r,　r → s,　s → q

三つの知識により，q，r，s が循環証明している. より簡単な例を挙げると，知識集合 Σ＝{p → q, q → p}とする. 次のページの図 3.10 で示されるように，同じ節点が繰り返す.

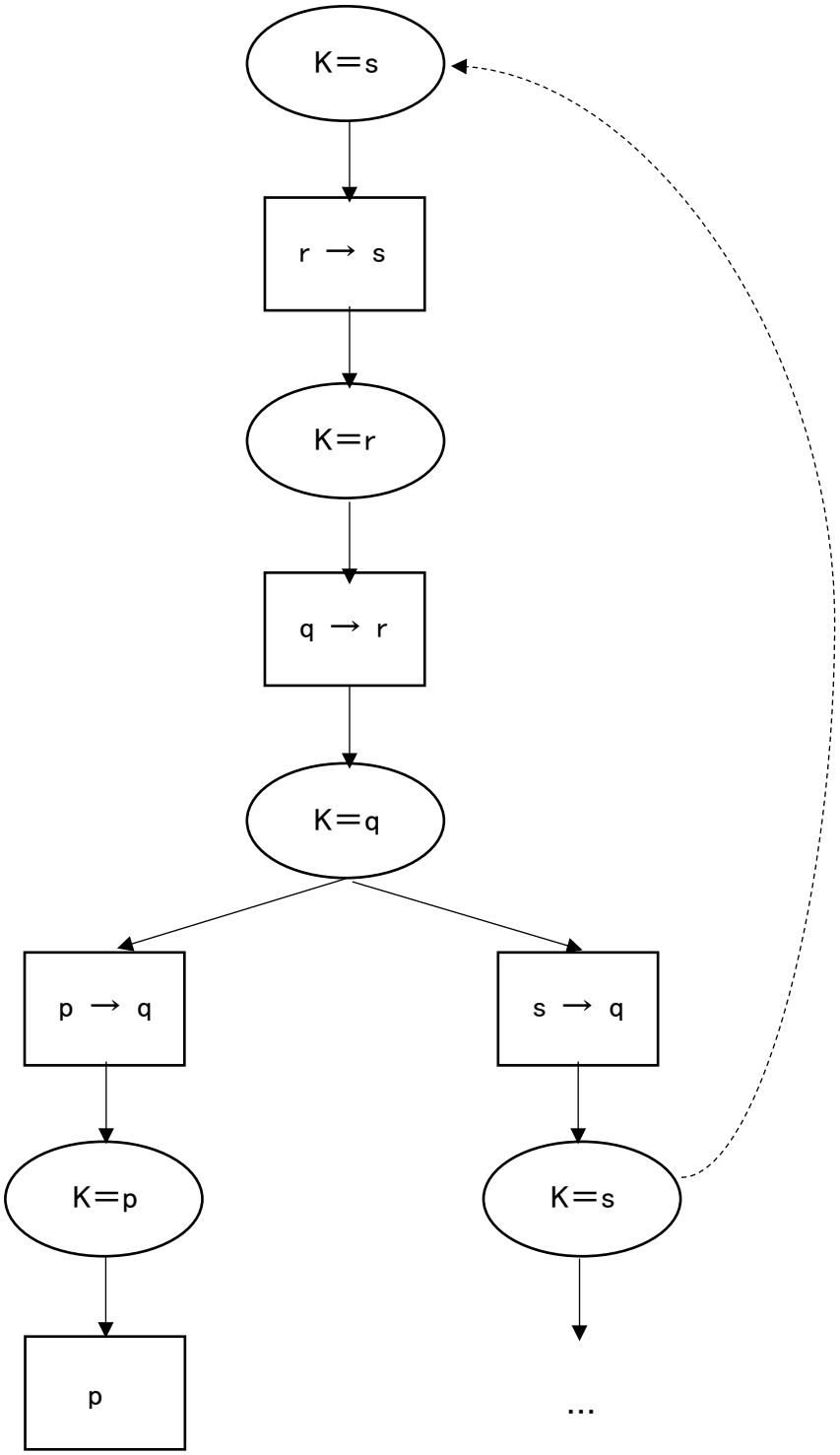


図 3.9　例 3.2 の無限木の生成過程

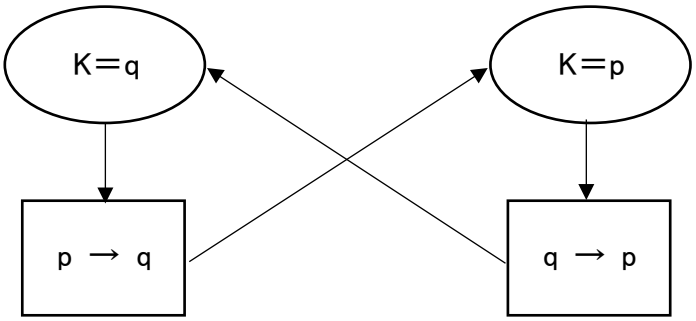


図 3.10　同じ節点が繰り返す例

このような木を生成しないように，次の解決案を考える．

無限木ができる理由は，先祖節点と構造が同様な節点が出現するからである．つまり，先祖節点と同じ節点から子節点を生成しなければよいということである．ここでは，節点にすべての先祖節点を記録することで，先祖節点と同様な節点であれば，この節点を葉節点とするという解決案を採用する．

3.2.3　木の節点の構造

以上のアルゴリズムにより，木の節点のデータを構造する．

木の節点に以下のデータを持つ．

① 節点のタイプ．結論節点かまたは知識節点か．
② 節点の原子論理式．結論節点であるときのみ有効である．
③ 節点の論理式．知識節点であるときのみ有効である．
④ 節点の先祖節点．3.2.2の問題を解決するため．ここでは，ある先祖節点と同様であれば子節点は持たないとする．
⑤ 節点の葉節点．
⑥ 節点の支持集合．アルゴリズムのステップ2で計算する．

3.2.4　木の生成の例

ここでは，少し複雑な例をあげ，木の生成過程を表す．

例 3.3　知識集合 Σ＝{p1, p2, p3, p4, p1 → q0, p1 → q1, p2 → q1, p2∧p3 → q2, p3 → ¬q3, p4 → ¬q3, p1 → q0, q0 → Q, q1 → Q, q2∧¬q3 → Q}，根節点の原子論理式は K＝Q である．図 3.11 で生成した木を表す．

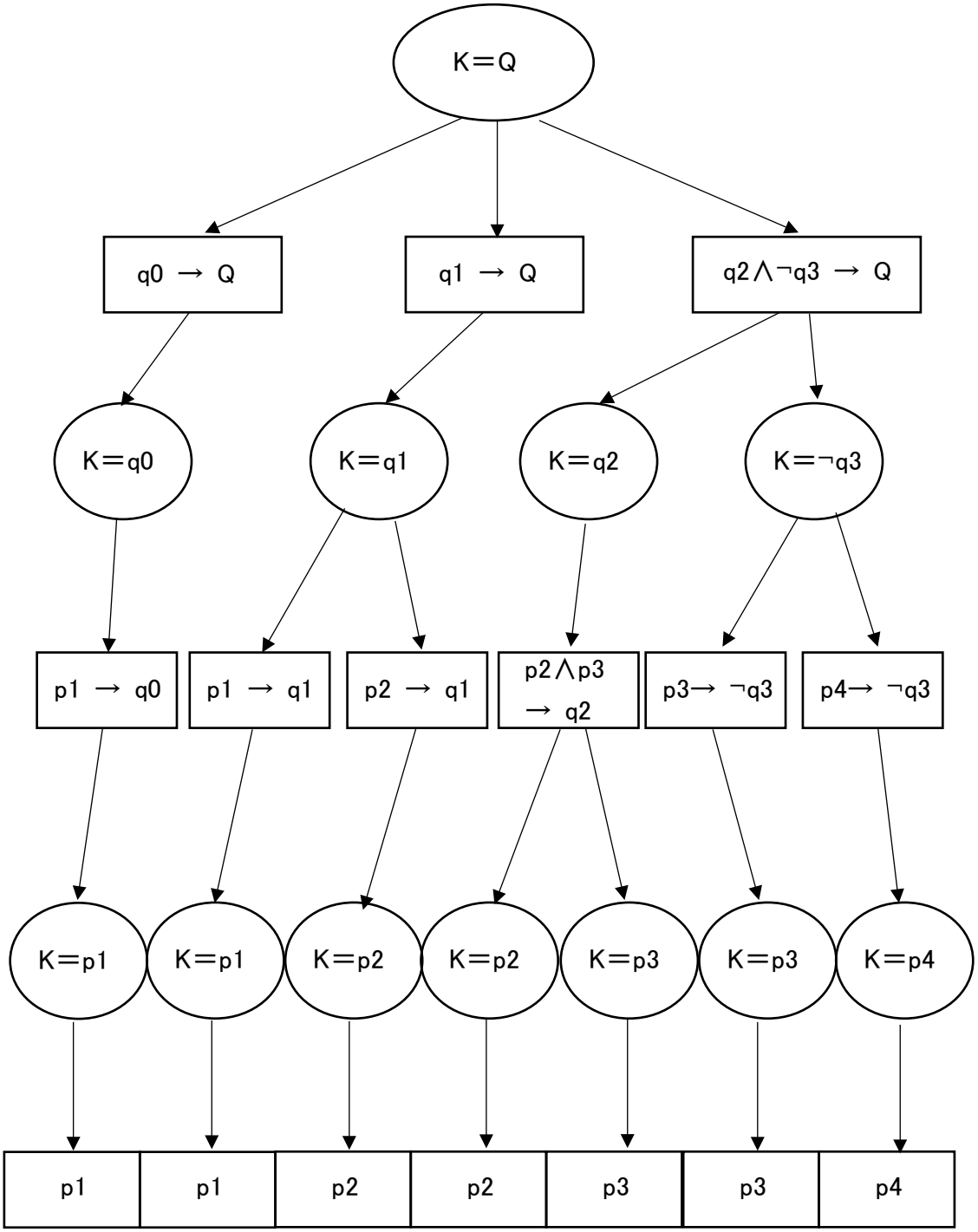


図 3.11　例 3.3 で生成した木

3.2.5　木から議論を得る

アルゴリズムの第二ステップは，木を用いて根節点の議論を得ることである．根節点における議論の結論は与えられたため，議論の支持を計算すればいい．また，議論が複数ある可能性があるので，ここでは支持の集合を求める．

本研究では，支持集合の計算を以下で定義する．

3.2.5.1　支持集合の計算

支持集合 $S=\{\Phi_1, \Phi_2, \cdots, \Phi_n\}$とする．

まず，支持と支持の計算を定義する．

定義 3.1　支持間の足し算を以下で定義する．

$\Phi_1=\{s_1, s_2, \cdots, s_m\}$

$\Phi_2=\{t_1, t_2, \cdots, t_n\}$とすると，

$\Phi_1 + \Phi_2=\{s_1, s_2, \cdots, s_m, t_1, t_2, \cdots, t_n\}$

であり，$\{s_1, s_2, \cdots, s_m, t_1, t_2, \cdots, t_n\}$が無矛盾であるとき，新たな支持が得られる．

また，$\{s_1, s_2, \cdots, s_m, t_1, t_2, \cdots, t_n\}$が無矛盾ではないとき，$\Phi_1 + \Phi_2= \varnothing$ と見なす．

特別に，Φ_2は空集合である時，$\Phi_1 + \Phi_2=\Phi_1$とする．

実際的意味では，議論 $A_1=\langle\Phi_1, \phi_1\rangle$と議論 $A_2=\langle\Phi_2, \phi_2\rangle$がある．$\Phi_1 + \Phi_2$が無矛盾であるとき，議論の定義から，$\langle\Phi_1 + \Phi_2, \phi_1 \wedge \phi_2\rangle$は議論であることがわかる．

次に，支持集合と支持集合の計算を二つ定義する．

定義 3.2　支持集合間の足し算を定義 3.1 と同様に以下で定義する．

$S=\{\Phi_1, \Phi_2, \cdots, \Phi_n\}$

$S'=\{\Phi_1', \Phi_2', \cdots, \Phi_m'\}$とすると，

$S+S'=\{\Phi_1, \Phi_2, \cdots, \Phi_n, \Phi_1', \Phi_2', \cdots, \Phi_m'\}$

であり，新たな支持集合が得られる．

特別に，S’が空集合であるとき，$S+S'=S$ である．

実際的意味では，支持集合 S 中の支持Φ_iと支持集合 S’中の支持Φ_j'（$1\leqq i\leqq n$, $1\leqq j\leqq m$）の結論は同様に ϕであるとき，支持集合 S＋S’中の支持の結論が全部 ϕであるということ．

さらに，支持集合と支持集合のかけ算を定義する．

定義 3.3　支持集合間のかけ算を以下で定義する.

$S=\{\Phi_1, \Phi_2, \cdots, \Phi_n\}$

$S'=\{\Phi_1', \Phi_2', \cdots, \Phi_m'\}$とすると,

$S\times S'=\{\Phi_1+\Phi_1', \Phi_1+\Phi_2', \cdots, \Phi_1+\Phi_m',$

$\Phi_2+\Phi_1', \Phi_2+\Phi_2', \cdots, \Phi_2+\Phi_m',$

$\cdots$

$\Phi_n+\Phi_1', \Phi_n+\Phi_2', \cdots, \Phi_n+\Phi_m'\}$

である. $S\times S'$が空集合でないとき, 新たな支持集合が得られる.

特別に, S'が空集合であるとき, $S\times S'$も空集合である.

実際的意味では, 支持集合 S 中の支持Φ_iの結論は ϕ_1 であり($1\leqq i\leqq n$), 支持集合 S'中の支持Φ_j'の結論は ϕ_2 であるとする($1\leqq j\leqq m$). 定義 3.1 により, $\Phi_i+\Phi_j'$が空集合でないとき, 支持$\Phi_i+\Phi_j'$では $\phi_1\wedge\phi_2$ という結論がある. すなわち, $S\times S'$が空集合でないとき, 支持集合内の支持の結論はすべて $\phi_1\wedge\phi_2$ である.

3.2.5.2　各節点が持つ支持集合の意義

すべての節点の構造内に支持集合というものを持つ. 結論節点と知識節点では, 支持集合の意味は異なる. それは, 前文で説明した節点を二種類分けた理由である.

結論節点が持つ支持集合が, 結論節点が持つ原子論理式を結論とする支持の集合である. また, 葉節点であるとき支持集合は空集合である.

知識節点が持つ支持集合が, 知識節点が持つ論理式の“→”の右を結論とし, 論理式が支持の要素の支持集合である. また, 葉節点であるとき, すなわち持つ論理式が原子論理式 ϕのとき, 支持集合は$\{\{\phi\}\}$である.

ここでは例を使ってより明白的に説明する.

例 3.4　節点が葉節点ではなく, 支持集合 $S=\{\Phi_1, \Phi_2, \cdots, \Phi_n\}$とする.

節点が結論節点であるとき, 持つ原子論理式を $K=\phi$とすると,

議論 $A_i=\langle\Phi_i, \phi\rangle(1\leqq i\leqq n)$が存在する.

節点が知識節点であるとき, 持つ論理式を $a_0, a_1, \cdots, a_n\rightarrow Q$ とすると,

議論 $A_i=\langle\Phi_i, Q\rangle$が存在する. ただし, $a_0, a_1, \cdots, a_n\rightarrow Q \in \Phi_i$ ($1\leqq i\leqq n$).

また, 節点が葉節点である時.

節点が結論節点であるとき, S が空集合である.

節点が知識節点であるとき, 持つ原子論理式を Q とすると,

$S=\{\{Q\}\}$であり, 議論 $A=\langle\{Q\}, Q\rangle$が存在する.

3.2.5.3　親節点が持つ支持集合と子節点が持つ支持集合の関係

ここでは，木の生成するとき親節点から子節点を生成する方法を用いる．また，親節点が結論節点かまたは知識節点か二つ問題を分ける分析する．

まず，親節点が結論節点であるとき，ここでは親節点が持つ原子論理式を K＝ϕとする．生成可能な子節点は以下の図で表す．

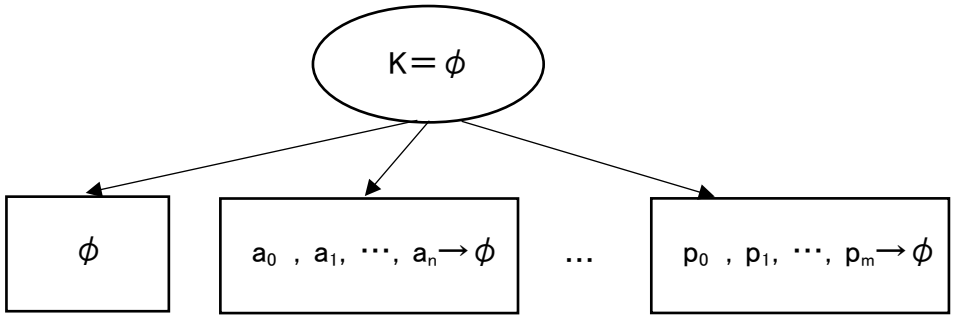


図 3.12　結論節点における生成可能な子節点

親節点の支持集合を S とし，子節点の支持集合を S_0, S_1，…S_pとすると，3.2.5.2 節で説明したように，議論 A_i＝<Φ_i，ϕ>が存在する．ただし，$\Phi_i \in S_j$．さらに，支持集合足し算の定義により，

$$S = S_0 + S_1 + \cdots + S_p$$

であることがわかる．

次に，親節点が知識節点であるとき，ここでは親節点が持つ論理式を

$$a_0 ,\ a_1,\ \cdots,\ a_n \rightarrow \phi$$

とする．生成可能な子節点は以下の図で表す．

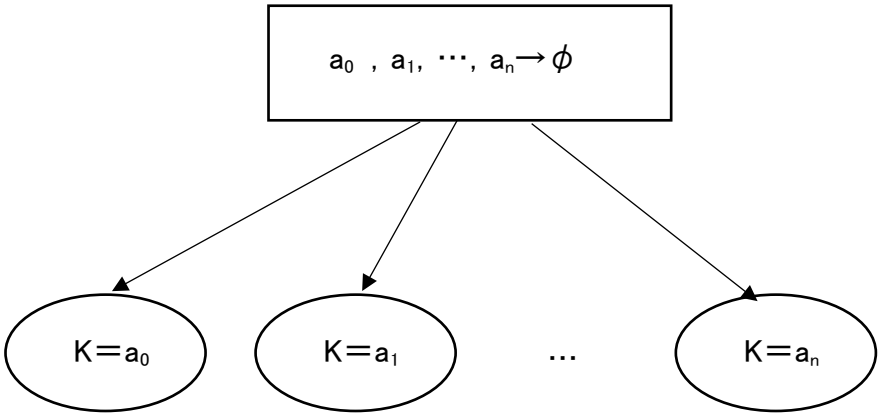


図 3.13　知識節点における生成可能な子節点

親節点の支持集合をSとし，子節点の支持集合をS_0, S_1，…S_nとすると，3.2.5.2 節で説明したように，議論 $A_i=\langle \Phi_i, a_i\rangle$が存在する．ただし，$\Phi_i \in S_i$．支持集合かけ算の定義により，$\forall \Phi \in S_0 \times S_1 \times \cdots \times S_n$のとき，議論$\langle \Phi, a_0 \wedge a_1 \wedge \cdots \wedge a_n\rangle$が存在する．さらに，
$\forall \Phi \in S_0 \times S_1 \times \cdots \times S_n \times \{\{a_0 , a_1, \cdots, a_n \rightarrow \phi\}\}$のとき，議論

$$\langle \Phi, a_0 \wedge a_1 \wedge \cdots \wedge a_n \wedge (a_0 , a_1, \cdots, a_n \rightarrow \phi)\rangle$$

が存在する．変換すると，議論$\langle \Phi, \phi\rangle$があり，また$(a_0 , a_1, \cdots, a_n \rightarrow \phi) \in \Phi$なため，支持集合 S とは同じである．よって，

$$S=S_0 \times S_1 \times \cdots \times S_n \times \{\{a_0 , a_1, \cdots, a_n \rightarrow \phi\}\}$$

であることがわかる．

葉節点持つ支持集合は木が生成するとき与えられたので，また，子節点の支持集合から親節点の支持集合を求める計算方法がわかった．よって，最終的に根節点の支持集合 $S=\{\Phi_1, \Phi_2, \cdots, \Phi_n\}$が得られる．根節点が持つ原子論理式を$K=\phi$とすると，議論集合 $T=\{A_1, A_2, \cdots, A_n\}$が得られる．ただし，$A_i=\langle \Phi , \phi\rangle$である$(1 \leqq i \leqq n)$．

ここでは例 3.3 の木からの計算の一部をあげて，子節点の支持集合から親節点の支持集合への計算方法を示す．

例 3.4　以下の部分木を考える．

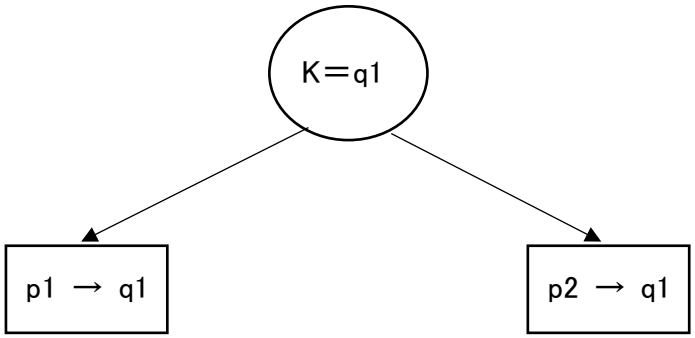


図 3.14　親節点が結論節点である部分木

ここで、子節点が持つ支持集合 S_1と S_2がそれぞれ以下で示す。

$$S_1=\{\{p1, p1 \rightarrow q1\}\}$$
$$S_2=\{\{p2, p2 \rightarrow q1\}\}$$

親節点が持つ支持集合 $S=S_1+S_2$である。定義により

$$S=S_1+S_2=\{\{p1, p1 \rightarrow q1\},$$
$$\{p2, p2 \rightarrow q1\}\}$$

となる。

例 3.5　以下の部分木を考える。

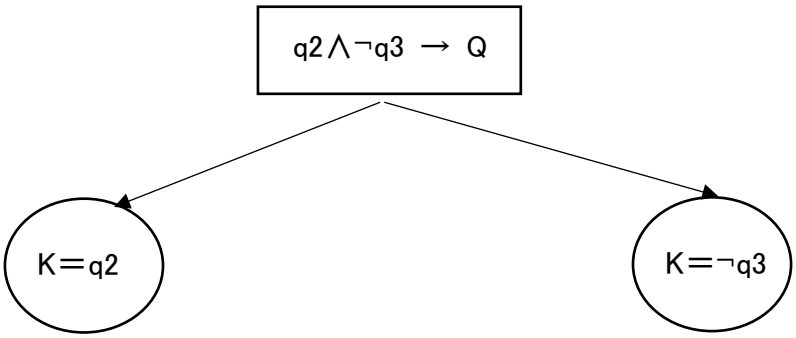


図 3.15　親節点が知識節点である部分木

ここで、子節点が持つ支持集合 S_1と S_2がそれぞれ以下で示す。

S_1＝{{p2, p3, p2∧p3 → q2}}

S_2＝{{p3, p3 → ¬q3},
{p4, p4 → ¬q3}}

である。

親節点が持つ支持集合 S＝S_1×S_2×{{q2∧¬q3 → Q}}である。定義により

S＝S_1×S_2×{{q2∧¬q3 → Q}}
＝{{p2, p3, p2∧p3 → q2, p3, p3 → ¬q3},
{p2, p3, p2∧p3 → q2, p4, p4 → ¬q3}}×{{q2∧¬q3 → Q}}
＝{{p2, p3, p2∧p3 → q2, p3, p3 → ¬q3, q2∧¬q3 → Q},
{p2, p3, p2∧p3 → q2, p4, p4 → ¬q3, q2∧¬q3 → Q}}

で計算できる。

例 3.6　以下の部分木を考える。

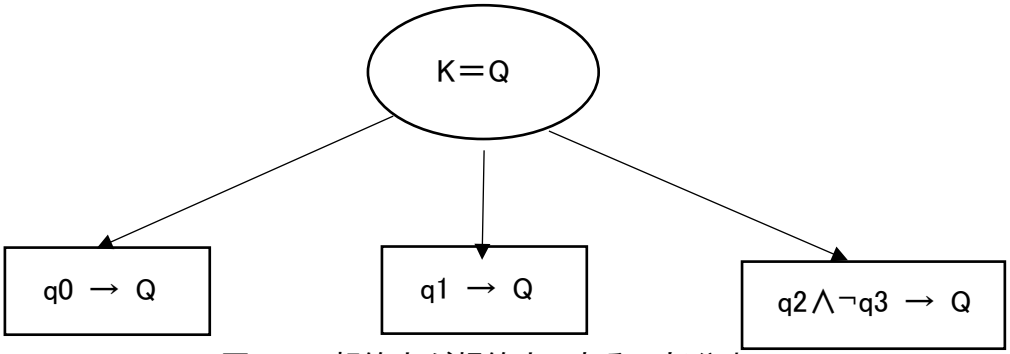


図 3.16　親節点が根節点であるの部分木

ここで、子節点が持つ支持集合 S_1と S_2と S_3がそれぞれ以下で示す。

S_1＝{{p1, p1 → q0, q0 → Q}}

S_2＝{{p1, p1 → q1, q1 → Q},
{p2, p2 → q1, q1 → Q}}

S_3＝{{p2, p3, p2∧p3 → q2, p3, p3 → ¬q3, q2∧¬q3 → Q},
{p2, p3, p2∧p3 → q2, p4, p4 → ¬q3, q2∧¬q3 → Q}}

である。

親節点が持つ支持集合 S＝S_1＋S_2＋S_3である。定義により

S＝S_1＋S_2＋S_3
＝{{p1, p1 → q0, q0 → Q},
{p1, p1 → q1, q1 → Q},
{p2, p2 → q1, q1 → Q}}＋S_3
＝{{p1, p1 → q0, q0 → Q},
{p1, p1 → q1, q1 → Q},
{p2, p2 → q1, q1 → Q},
{p2, p3, p2∧p3 → q2, p3, p3 → ¬q3, q2∧¬q3 → Q},
{p2, p3, p2∧p3 → q2, p4, p4 → ¬q3, q2∧¬q3 → Q}}

で計算する。

ここで, S は根節点が持つ支持集合であるため, アルゴリズムにおける出力でもある。

例 3.3 をプログラムで計算すると, 結果が図 3.13 で示される.

図 3.17 プログラムで得た支持集合

ここでは, 一行を一つの支持を表す. また, 第四行には p3 が二つあるが, それは, p3 という知識が二回使ったことを意味する.

アルゴリズムにおける出力である支持集合 S の実質の意味では, 支持集合 S のすべての要素から, Q という結論を導ける。すなわち

∀Φ ∈ S ， A＝<Φ, Q>は議論である

また、議論の概念を用いた人工検査では，出力 S が正解であり，プログラム上の問題がないということがわかる。

3.3　分析対象の推理小説の紹介

本研究では，Baroness Orczy（バロネス・オルツィ）の推理小説集『The Old Man in the Corner（隅の老人）』中の推理小説「The Theft at the English Provident Bank（〈イギリス共済銀行〉強盗事件）」[9]を対象として分析する．また，この推理小説で登場した探偵は安楽椅子探偵と呼び，事件そのものとは無関係であるため，事件の登場人物ではないということになる．

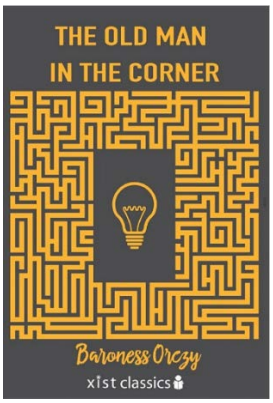


図 3.18　小説集『The Old Man in the Corner（隅の老人）』[9]

3.3.1　小説の内容紹介

ここでは，まず小説「The Theft at the English Provident Bank」（〈イギリス共済銀行〉強盗事件）の内容を紹介する．

事件の登場人物は以下である．ここでは名前ではなく，身分で登場人物を表示する．

- 銀行の支配人
- 支配人の妻
- 支配人の息子
- 銀行の警備員

次に，この小説における事件の内容を紹介する．

銀行の支配人は彼の妻と息子とともに銀行の上の部屋で生活していた．夜になると，銀行の警備員は外の人が銀行に入れないように銀行の金庫を守っている．

ある日の朝，支配人がショックで重病になった．それは，前の夜に誰かが銀行の金庫から金を盗んだからだ．警察の調査では，もし盗賊が外部の人であるなら，彼が金庫に入ると必ず銀行の警備員に気づかれる．また，金庫に入るために，金庫の鍵はなくてはならないものである．それを手に入れるのは，支配人の家族だけである．

銀行の警備員によると，事件の夜，外から金庫に入った人は一人だけだそうだ．また，警備員の証言では，彼がその人は誰かが見えなかったが，支配人の妻はそれが支配人であると言ったので，自分もその人は支配人だと思ったそうだ．

しかし，支配人の妻は，事件の夜，自分は警備員に会ったことがないと言った．さらに，彼女は金庫に入った人が支配人であるということを言った記憶もない．

また，支配人の息子の証言では，事件の後，彼は父とともに家に戻った．それ以外は何も知らないそうだ．

後日，支配人は，事件の夜，自分はコンサートに行ったと言った．そして，会場のスタッフも支配人のアリバイを証明した．

また，それ以外の証拠から，支配人の妻は犯人ではないということと，犯人が一人であることがわかった．

さて，金庫から金を盗んだ犯人は誰だ．

3.3.2　事件内容の知識化

本研究では，推理小説における事件の内容を分析し，取り扱うことができる知識に変換する方法を，事件内容の知識化と呼ぶ．また，推理小説における推理に必要な知識を「証拠」，「証言」と「常識推論」三つ分けて考える．

「証拠」とは，小説における作者が明示した必ず真実である知識のことである．一般的に，推理小説における証拠というのは，地の文で書かれた知識，または警察が公表された調査結果などである．

「証言」とは，登場人物が事件についての発言である．本研究において，一つの証言を2つの要素に分けて考える．一つは，誰かが証言したということである．もう一つは，その証言で何が示されたかということである．その理由は，証言者は嘘をつかなければ，その

証言は真実であるが，嘘をつくときでも，その証言は偽であるとは限らないからである．

「常識推論」とは，証拠と証言から何かを導くための知識である．あるいは，人間が持つ推理能力を表現する知識でも言える．常識推論がないと，証拠と証言から新たな知識を導くことができない．しかし，常識であるとしても，場合によって正しくないこともある．もっとも，読者が推理する際に，間違う結論を出す原因は，正しく見える常識が実際は思い込みであることが大数ある．

推理小説中の知識を「証拠」「証言」「常識推論」三つ分けることにより，より系統的に情報を取り上げることができる．しかし，本研究では自然言語処理を使用しないため，原文の情報から知識へ変換する際に恣意的になる可能性がある．また，各命題の持つ情報の粒度における妥当な大小を探ることも問題の一つとなる．

その問題を解決するため，本研究では推理小説の中の人物の推理を利用する．

一般的に，推理小説中真相を推理する人物は少なからず二人存在する．例えば，『シャーロック・ホームズ』シリーズ中のホームズとワトソン，または本研究の研究材料である『隅の老人』中の老人と女性新聞記者のような推理者がいる．ここでは彼らの推理に基づいて命題を取り出す。何の結論を出すために、推理する人物の常識（または思い込み）と何の証拠が必要である、という形で各命題を決める。このように決められた命題が持つ情報は小説中の推理者が推理する際に採用した知識であるため，粒度における大きさは適切であると考えられる．

3.3.3　小説中の推理者の推理

本研究における分析対象である小説「The Theft at the English Provident Bank」（〈イギリス共済銀行〉強盗事件）では，探偵役である老人と女性新聞記者が事件の真相を推理していた．それらを用いた幾つの推論の例を以下で示す．なお，ここでは探偵の推理が正解であるという考えで影響されるため，推論する人物は表せない．

支配人はかなり大きなショックを受けた．もし彼が犯人であれば，ショックを受けることがない．よって，支配人が犯人ではない．

もし事件の夜に，支配人の妻の証言し，その証言も正しくてあれば，そのとき支配人が金庫にいた．

警備員の証言が正しくてあれば，事件の夜に支配人の妻が証言した．

金庫に入るため鍵が必要なので，犯人は必ず支配人または彼の家族である．

現在支配人の妻の証言が正しくてあれば，警備員の証言が嘘になる.

……

こういった推論により，証拠，証言，常識推論とそれらの論理式表現が以下の表で表せる.

証拠	論理式
支配人がショックを受けた	managerShock
金庫に入れるのは，支配人と彼の家族だけである	only_family_can_enter
妻は犯人ではない	¬Wife_Criminal

表 3.1　証拠

証言	論理式
事件の夜，支配人の妻の証言によれば，支配人は金庫にいた	Wife_T1→manager_was_there
警備員が証言する	Guide_T
警備員の証言によれば，事件の夜，支配人の妻が証言した	Guide_T→Wife_T1
現在支配人の妻が証言した	Wife_T2
現在支配人の妻の証言によれば，事件の夜に彼女がなにも証言しない	Wife_T2→ ¬Wife_T1
コンサートのスタッフが証言した	Stuff_T
スタッフの証言によると，支配人にはアリバイがある	Stuff_T→AlibiManager

表 3.2　証言

常識推論	論理式
ショックを受けるが犯人ではない	managerShock→¬ManagerCriminal
事件の夜に現場がいた人が犯人である	manager_was_there→ManagerCriminal
事件の夜，支配人の妻がなにも証言しなかったら，支配人は金庫にいなかった	¬Wife_T1→¬manager_was_there
事件の夜に現場がいなかった人が犯人ではない	¬manager_was_there→¬ManagerCriminal
アリバイのある人は犯人ではない	AlibiManager→¬ManagerCriminal
犯人は金庫に入れる人である	only_family_can_enter→FamilyCriminal
消去法	FamilyCriminal ， ¬ManagerCriminal ， ¬WifeCriminal→SonCriminal

表 3.3 常識推論

3.3.4 実験結果

ここでは，犯人を関係する論理式は ManagerCriminal, WifeCriminal, SonCriminal 三つである. そのため、アルゴリズムにより、結論がそれぞれ ManagerCriminal, WifeCriminal, SonCriminal, ¬ManagerCriminal, ¬WifeCriminal, ¬SonCriminal である議論の支持の可能集合を計算し，以下で表す.

結論が ManagerCriminal である支持

図 3.19 結論が ManagerCriminal である支持

{ Guide_T ， Guide_T → Wife_T1 ， Wife_T1 → manager_was_there ， manager_was_there → ManagerCriminal}

結論が¬ManagerCriminal である支持

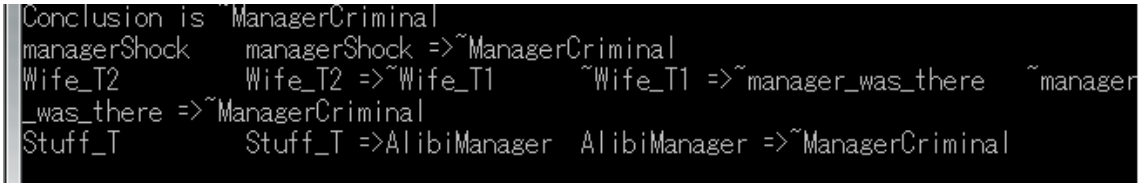


図 3.20 結論が¬ManagerCriminal である支持

{managerShock, managerShock→¬ManagerCriminal}

{Wife_T2, Wife_T2→ ¬Wife_T1, ¬Wife_T1→¬manager_was_there, ¬manager_was_there→¬ManagerCriminal}

{Stuff_T, Stuff_T→AlibiManager, AlibiManager→¬ManagerCriminal}

結論が WifeCriminal である支持

```
Conclusion is WifeCriminal
NO SHIJI
```

図 3.21　結論が WifeCriminal である支持

{}

結論が¬WifeCriminal である支持

```
Conclusion is ~WifeCriminal
~WifeCriminal
```

図 3.22　結論が¬WifeCriminal である支持

{¬WifeCriminal}

結論が SonCriminal である支持

```
Conclusion is SonCriminal
only_family_can_enter   only_family_can_enter =>FamilyCriminal  managerShock
managerShock =>~ManagerCriminal         ~WifeCriminal   FamilyCriminal ~ManagerC
riminal ~WifeCriminal =>SonCriminal
only_family_can_enter   only_family_can_enter =>FamilyCriminal  Wife_T2
Wife_T2 =>~Wife_T1      ~Wife_T1 =>~manager_was_there   ~manager_was_there =>~Ma
nagerCriminal   ~WifeCriminal   FamilyCriminal ~ManagerCriminal ~WifeCriminal =>
SonCriminal
only_family_can_enter   only_family_can_enter =>FamilyCriminal  Stuff_T
Stuff_T =>AlibiManager  AlibiManager =>~ManagerCriminal         ~WifeCriminal
FamilyCriminal ~ManagerCriminal ~WifeCriminal =>SonCriminal
```

図 3.23　結論が SonCriminal である支持

{ managerShock , managerShock → ¬ManagerCriminal , ¬WifeCriminal , FamilyCriminal , ¬ManagerCriminal, ¬WifeCriminal→SonCriminal}

{Wife_T2, Wife_T2→ ¬Wife_T1, ¬Wife_T1→¬manager_was_there, ¬manager_was_there→ ¬ManagerCriminal , ¬WifeCriminal , FamilyCriminal , ¬ManagerCriminal , ¬WifeCriminal → SonCriminal}

{ Stuff_T , Stuff_T → AlibiManager , AlibiManager → ¬ManagerCriminal , ¬WifeCriminal , FamilyCriminal, ¬ManagerCriminal, ¬WifeCriminal→SonCriminal}

結論が¬SonCriminal である支持

```
Conclusion is ~SonCriminal
NO SHIJI
```

図 3.24 結論が¬SonCriminal である支持

{}

システムの出力は以上である.

ここでは,「妻が犯人である」を結論とする支持がなく,「妻が犯人ではない」を結論とする支持が持つため, 小説から「妻が犯人ではない」という結論を導くことを示す.

また,「支配人が犯人である」と「支配人の息子が犯人である」を表す論理式を結論とする議論がともにある. しかし,「支配人が犯人ではない」を表す論理式を結論とする議論が存在するが,「支配人の息子が犯人である」を表す論理式を結論とする議論が存在しない.

以上の結果から, この推理小説では,「支配人が犯人である」と「支配人の息子が犯人である」を結論とする推理がともにあるが,「支配人が犯人である」を反駁する推理があるが,「支配人の息子が犯人である」を反駁する推理が存在しないということである. これを現実の議論を例えると, 二人が異なることを主張しているか、一人が相手の主張を反駁できるか, もう一人が反駁することができない. この状況における論争の勝利者はもちろん, 相手の主張を反駁できる側である.

よって, 本システムの最終出力である犯人は,「支配人の息子」である.

3.3.5　評価

システムの最終出力である「支配人の息子」が小説における老人の推理結果とは同じである. よって, この推理小説において, 本システムは有効であることが示される.

第4章　終わりに

4.1　まとめ

本研究では，人間の論争をモデルで発展した議論理論について紹介し，それを用いて，矛盾ある知識集合の取り扱いが可能になることを説明する．また，推理小説の性質より，示された知識を矛盾ある知識集合として扱うことができることを説明する．以上のことにより，本研究では議論理論を用いて犯人を推理システムの実装および評価を目標としていた．

次に，推理システムにおけるアルゴリズムを説明し，また，小説集『The Old Man in the Corner（隅の老人）』中の推理小説「The Theft at the English Provident Bank（〈イギリス共済銀行〉強盗事件）」[9]を対象として、システムの流れを説明する．さらに，システムの出力を説明し，小説内容の比較によってシステムを評価する．

4.2　今後の課題

今後の課題について，以下の問題が残っている．

- 本研究における推理小説中の知識表現では命題論理を用いている．述語論理における知識と比較すると，具体的な表現力が低いところがある．それで，知識を表現するため，多くの命題論理式が必要となる．また，似ている知識の表現間では，その相関性を表すのが困難である．例えば，「犯人がＡである」と「犯人がＢである」二つの知識が命題論理で関係性を表したいのなら，新たな知識を加えるのが必要であるが，述語論理では，そのままでも二つの知識の関係を表せる．そのため、今後の課題の一つとして，議論理論における知識の表現を述語論理へ拡張し，それを用いて推理小説中の知識を表現し推理する．
- 本研究では，自然言語処理による解析が行わず，人工的に推理小説から知識を取り出すため，知識の分類や，小説中推理者の推理を用いることなどの方法で，知識の客観性を保ちたい．しかし，どちらの方法にも限界があるため，知識表現が不自然なところがある．また，その知識の客観性についても評価しがたい．そのため，自然

言語処理の解析を用いる推理小説からの推理システムも今後の課題としてあげられる.

参考文献

[1] 太原育夫,延澤志保. ″矛盾した知識集合からの推論.″ 電子情報通信学会論文誌 D 87.10 (2004): 931-938.

[2] 向後英二,亀田弘之. ″矛盾とその処理方法に関する一考察.″ 電子情報通信学会技術研究報告. TL, 思考と言語 99.691 (2000): 17-24.

[3] Besnard, Philippe, and Anthony Hunter. ″Towards a logic-based theory of argumentation.″ AAAI/IAAI. 2000.

[4] Dung, Phan Minh. ″On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games.″ Artificial intelligence 77.2 (1995): 321-357.

[5] 若木利子, 新田克己,『数理議論学』, 東京電機大学出版局, 2017.

[6] Amgoud, Leila, and Claudette Cayrol. ″A reasoning model based on the production of acceptable arguments.″ Annals of Mathematics and Artificial Intelligence 34.1 (2002): 197-215.

[7] Bench-Capon, Trevor JM. ″Persuasion in practical argument using value-based argumentation frameworks.″ Journal of Logic and Computation 13.3 (2003): 429-448.

[8] Toni, Francesca. ″A tutorial on assumption-based argumentation.″ Argument & Computation 5.1 (2014): 89-117.

[9] Baroness Orczy.「The Old Man in the Corner」, Xist Publishing, 2015.