論理と推論

白井 良明
立命館大学情報理工学部
知能情報学科
shirai@ci.ritsumei.ac.jp

命題論理

素命題: P, Q, ...
論理記号: ∧, ∨, ~, →
命題: P ∧ Q, P ∨ Q, ~P, P → Q, P ≡ Q

P ∧ Q :**論理積、連言**、and
P ∨ Q **を論理和、選言**、or
~P : **否定**、not
P → Q: **含意**、implication
Q ≡ Q: **同値**、equivalence

P → Q ≡ ~P ∨Q

P	Q	P ∧ Q	P ∨ Q	P → Q	P ≡ Q
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

述語論理

ON(monkey, box)　ON: 述語記号(predicate symbol) T or F、
monkey, box: 定数記号
AT(monkey, x)　x: 変数記号
child(x): x の子供　child: 関数記号

述語論理

ON(monkey, box)　ON: 述語記号(predicate symbol) T or F、
monkey, box: 定数記号
AT(monkey, x)　x: 変数記号
child(x): x の子供　child: 関数記号

結合記号： $\wedge, \vee, \sim, \rightarrow, \equiv$
量記号： $\forall$（全称記号）、$\exists$（存在記号）
項(term): 定数記号、変数記号、関数、T, F
素命題： 述語記号と項から成る。
例： ON(x, y), AT(child(x), a)

述語論理

ON(monkey, box)　ON: **述語記号**(predicate symbol) T or F、
monkey, box: **定数記号**
AT(monkey, x)　x: **変数記号**
child(x): x **の子供**　child: **関数記号**

結合記号：$\wedge, \vee, \sim, \rightarrow, \equiv$
量記号：　$\forall$ (**全称記号**)、$\exists$ (**存在記号**)
項(term): **定数記号、変数記号、関数、** T, F
素命題：　**述語記号と項から成る。**
例：　ON(x, y), AT(child(x), a)
命題 (well-formed-formula, wff): **素命題、命題を結合記号でつないだもの、変数を量記号で指定したもの**
P,Q**を命題とすると、** $P \wedge Q$, $P \vee Q$, $\sim P$, $P \rightarrow Q$, $P \equiv Q$
$(\forall x)[(\exists y)$ LESS(x, y)], $\forall x \exists y$ LESS(x, y),
$\exists x \forall y$ LESS(x, y)

述語論理

命題の例: $\forall x \forall y \forall z$ [LESS(x, y) $\wedge$ LESS(y, z) $\rightarrow$LESS(x, z)]

妥当命題: **述語記号にどのような解釈を与えても真**
Valid wff
$\forall x$ P(x) $\rightarrow$P(a)

充足不能命題:**述語記号にどのような解釈を与えても偽**
Unsatisfiable wff
$\forall x$ P(x) $\rightarrow \sim$P(a)

節形式

リテラル(literal): **素命題、素命題の否定**
節形式(clause) : **リテラル、リテラルの選言**

$\forall x\ \forall y\ \forall z\ [P \wedge Q \wedge R]$　　　　[ ]**内を母式**(matrix)

節形式

リテラル(literal): **素命題、素命題の否定**
節形式(clause) : **リテラル、リテラルの選言**

$\forall x\ \forall y\ \forall z\ [P \wedge Q \wedge R]$　　　　[ ]**内を母式**(matrix)

節形式への変換

1. $P \rightarrow Q$ **を** $\sim P \vee Q$, $P \equiv Q$ **を**$(\sim P \vee Q) \wedge (\sim Q \vee P)$
2. $\forall x[\sim P(x) \vee \exists x Q(x)]$ **を** $\forall x[\sim P(x) \vee \exists y Q(y)]$
3. $\sim \forall x\ [P(x) \wedge \sim (Q(x) \wedge R(x))]$ **を** $\exists x\ [\sim P(x) \vee (Q(x) \wedge R(x))]$

節形式

リテラル(literal): **素命題、素命題の否定**
節形式(clause) : **リテラル、リテラルの選言**

$\forall x \, \forall y \, \forall z \, [P \land Q \land R]$　　　　[ ]**内を母式**(matrix)

節形式への変換

1. $P \to Q$ **を** $\sim P \lor Q$, $P \equiv Q$ **を** $(\sim P \lor Q) \land (\sim Q \lor P)$
2. $\forall x[\sim P(x) \lor \exists x Q(x)]$ **を** $\forall x[\sim P(x) \lor \exists y Q(y)]$
3. $\sim \forall x \, [P(x) \land \sim (Q(x) \land R(x))]$ **を** $\exists x \, [\sim P(x) \lor (Q(x) \land R(x))]$
4. $\forall x[\exists y \, P(x, y)]$ **を** $\forall x[P(x, f(x))]$　Skolem-function
   $\exists x \, P(x)$ **を** $P(a)$

節形式

リテラル(literal): **素命題、素命題の否定**
節形式(clause) : **リテラル、リテラルの選言**

$\forall x \, \forall y \, \forall z \, [P \land Q \land R]$　　　　[ ]**内を母式**(matrix)

節形式への変換

1. $P \to Q$ **を** $\sim P \lor Q$, $P \equiv Q$ **を** $(\sim P \lor Q) \land (\sim Q \lor P)$
2. $\forall x[\sim P(x) \lor \exists x Q(x)]$ **を** $\forall x[\sim P(x) \lor \exists y Q(y)]$
3. $\sim \forall x \, [P(x) \land \sim (Q(x) \land R(x))]$ **を** $\exists x \, [\sim P(x) \lor (Q(x) \land R(x))]$
4. $\forall x[\exists y \, P(x, y)]$ **を** $\forall x[P(x, f(x))]$　Skolem-function
   $\exists x \, P(x)$ **を** $P(a)$
5. $\forall x[P(x) \lor \forall y \sim Q(y)]$ **を** $\forall x \, \forall y \, [P(x) \lor \sim Q(y)]$
6. $(A \land B) \lor (B \land C)$ **を** $(A \lor B) \land (A \lor C) \land B \land (B \lor C)$

推論の基本

1．P と P → Q　からQを導く。

P と ~P ∨Q から Q

2．P → Q と Q → R　から P → R を導く。

~P ∨Q と ~Q ∨R から ~P ∨R

一般に、

$C_1 = P \vee Q_1 \vee Q_2 \vee \ldots \vee Q_m$ と $C_2 = \sim P \vee R_1 \vee R_2 \vee \ldots \vee R_n$ から

$C_r = Q_1 \vee Q_2 \vee \ldots \vee Q_m \vee R_1 \vee R_2 \vee \ldots \vee R_n$ を導く。

C_1 と C_2 を親節、C_r を導出節(resolvent clause)という。

Procedure unify (P_1 ,P_2)

1 s:=NIL　　$Q_1 := P_1$,　$Q_2 := P_2$

2 LOOP: $D_i = Q_1$ とQ_2の不一致集合

3 **if** empty(D) then return (s)

4 t_1:=first(D), t_2:=last(D)

5 **if** variable(t_1) and not-contain(t_1 , t_2)
　　then s1: = (t1 / t2)
　　else if variable(t2) and not-contain(t2 , t1)
　　　　then s1: =(t2 / t1)
　　else return(fail)

6 $Q_1 := Q_1 s_1$, $Q_2 := Q_2 s_1$, s:=s s_1

7 go to LOOP

導出原理による証明

前提： $\forall x \exists v\, P(x,v) \wedge \forall y \forall z [P(a,y) \to \sim Q(y,z)]$
目標：$\sim \forall u \exists w Q(u,w)$

目標の否定： $\forall u \exists w Q(u,w)$
節形式： $Q(u, g(u))$

前提の節形式：

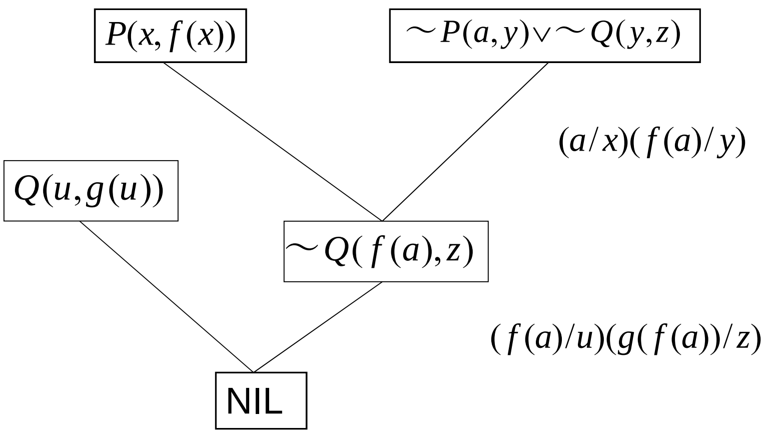
$$P(x, f(x))$$
$$\sim P(a,y) \vee \sim Q(y,z)$$

導 出 過 程

目標の否定： $Q(u,g(u))$

前提：

$$P(x, f(x))$$
$$\sim P(a,y) \vee \sim Q(y,z)$$



証明の制御

目標の否定　$\sim P(x,y) \vee Q(x,y)$

前提

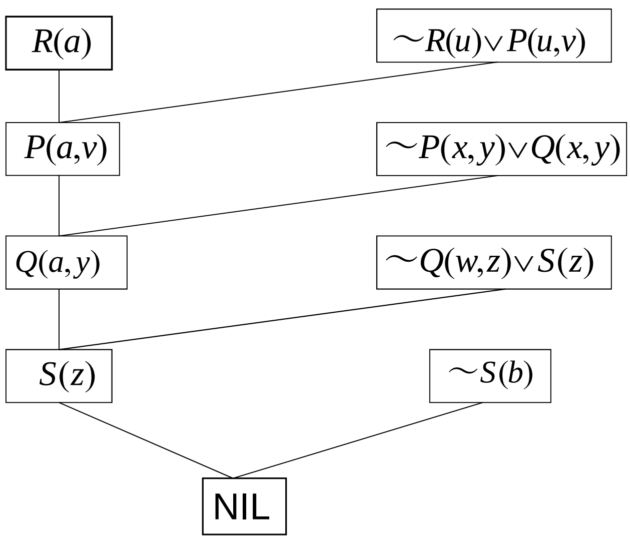
$\sim R(u) \vee P(u,v)$

$\sim Q(w,z) \vee S(z)$

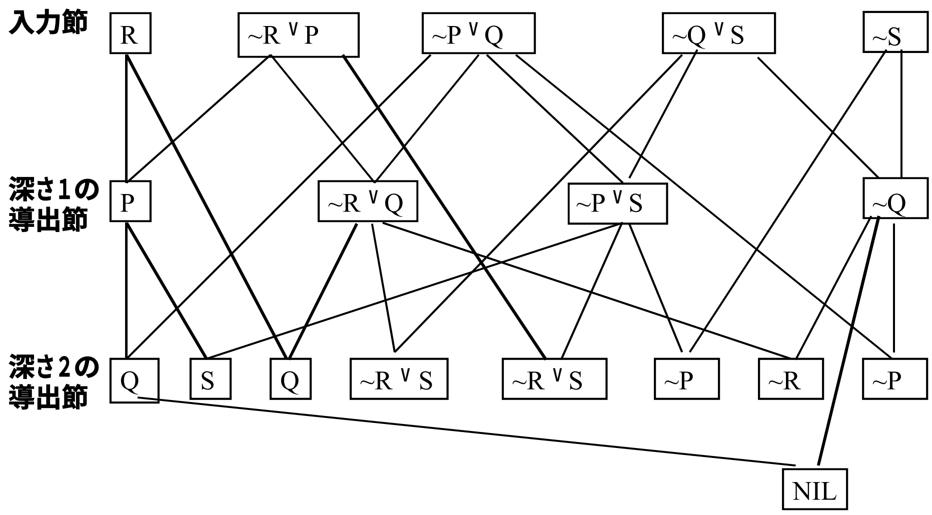
$R(a)$

$\sim S(b)$

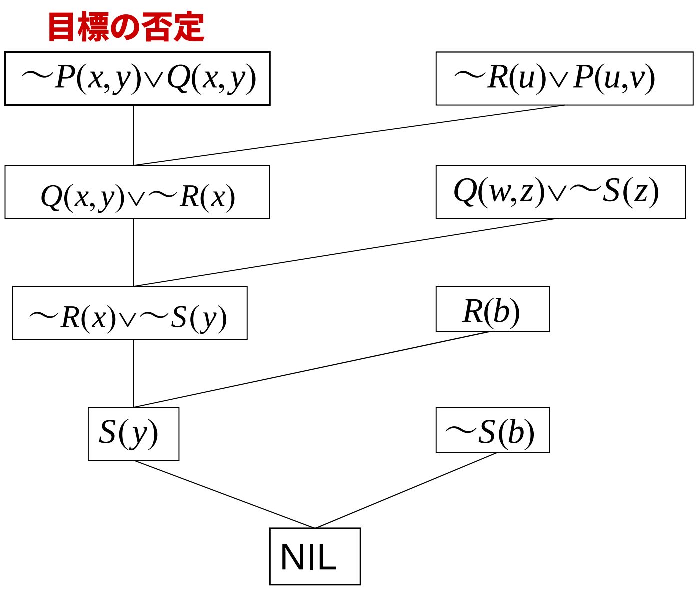
線形戦略による導出過程



幅優先戦略による導出過程



支持集合戦略による導出過程



答えの表現

目標の否定　$\sim P(x,y) \vee Q(x,y) \vee (P(x,y) \wedge \sim Q(x,y))$

前提

$\sim R(u) \vee P(u,v)$

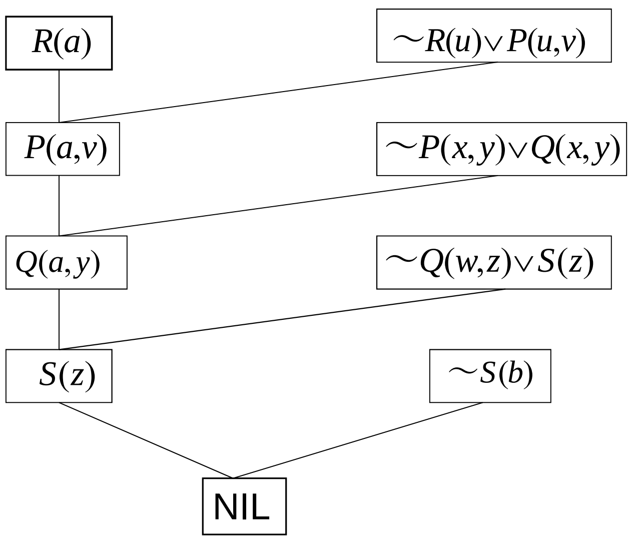
$\sim Q(w,z) \vee S(z)$

$R(a)$

$\sim S(b)$

目標　$(P(x,y) \wedge \sim Q(x,y))$

線形戦略による導出過程



線形戦略による導出での答の表現

$R(a)$　　$\sim R(u) \vee P(u,v)$

$P(a,v)$　　$\sim P(x,y) \vee Q(x,y) \vee (P(x,y) \wedge \sim Q(x,y))$

$Q(a,y) \vee (P(a,y) \wedge \sim Q(a,y))$　　$\sim Q(w,z) \vee S(z)$

$S(z) \vee (P(a,z) \wedge \sim Q(a,z))$　　$\sim S(b)$

$NIL \quad \vee (P(a,b) \wedge \sim Q(a,b))$

目標：$\sim \forall u \exists w Q(u,w)$

答の表現の具体例

命題：誰にも祖母父がいる

$\forall x \exists y G(x,y)$

命題の否定：

$\exists x \forall y \sim G(x,y) \equiv \sim G(a,y)$

誰にも親がいる　$P(z, f(z))$

祖父母の定義

$P(u,v) \wedge P(v,w) \rightarrow G(u,w)$

親

$\sim G(a,y)$　　$\sim P(u,v) \vee \sim P(v,w) \vee G(u,w)$

$a/u,\ w/y$

$\sim P(a,v) \vee \sim P(v,w)$　　$P(z, f(z))$

$a/z,\ f(a)/v$

$\sim P(f(a),w)$　　$P(z, f(z))$

$f(a)/z,\ f(f(a))/w$

NIL

答の表現の具体例

命題：誰にも祖父母がいる

$\forall x \exists y G(x,y)$

命題の否定：　$\sim G(a,y)$

命題

$\sim G(a,y) \vee G(a,y)$

$\sim P(u,v) \vee \sim P(v,w) \vee G(u,w)$

$a/u,\ w/y$

$\sim P(a,v) \vee \sim P(v,w) \vee G(a,w)$

$P(z, f(z))$

$a/z,\ f(a)/v$

$\sim P(f(a),w) \vee G(a,w)$

$P(z, f(z))$

$f(a)/z,$

$f(f(a))/w$

$G(a, f(f(a)))$

第一階述語論理（ＦＰＣ）

「私は本を持つ」

$$\exists x[have(I,x) \wedge book(x)]$$

「私は本かノートを持つ」

$$\exists x[have(I,x) \wedge book(x)] \vee \exists x[have(I,x) \wedge notebook(x)]$$

「すべての女性はケーキが好きだ」

$$\forall x[girl(x) \rightarrow \exists y[loves(x,y) \wedge cake(y)]]$$

「誰もそれをできない」

$$\neg \exists x[human(x) \wedge do-it(x)]$$

「ペンギン以外の鳥は飛ぶ」

$$\forall x[bird(x) \wedge \neg penguin(x) \rightarrow fly(x)]$$

NL $\longrightarrow$ parser $\longrightarrow$ FPC $\longrightarrow$ Database

非単調論理による推論

公理の集合 A から導かれる定理の集合：Th (A)

$$A \subset B \Rightarrow Th(A) \subseteq Th(B)$$

これを論理体系の単調性という

単調論理 ←→ 非単調論理

1 閉世界仮説

2 サーカムスクリプション

3 デフォールト推論

デフォールト推論 (default reasoning)

P が成立ち、$\sim$*Q* が証明されていないならば、
Q が成り立つ.

$$\frac{P : \mathrm{M}Q}{Q}$$

これは正規デフォールト規則
P を前提、*Q* を仮定、*R* を結論とよぶ.

$$\frac{BIRD(x) : MFLY(x)}{FLY(x)}$$

"Most birds fly"という意味

正規デフォールト体系

命題 (well-formed-formula) の集合 W と
正規デフォールト規則の集合 D からの
論理体系 (W, D) を正規デフォールト体系,

デフォールト論理体系から得られる命題の集合を
拡張 (extension) とよぶ。

デフォールト規則: Most birds fly と、

W＝ ~PENGUIN(x) V ~FLY(x),

BIRD(a), BIRD(b), PENGUIN(b)

から構成される論理体系の拡張Eは

E＝ W, FLY(a)

デフォールト推論のアルゴリズム

1. W と CON (D) から g を証明する。

S:＝W

2. Ds:＝証明に用いたDの規則の集合
If Ds＝{}, go to 4.
Else S:＝S U CON (Ds)

3. W と CON (D) からPRE (Ds) を証明する。
go to 2

4. Sが充足可能であることを示す。

非 単 論 論 理

推論システム

デフォールト推論

サーカムスクリプション

推論結果

信念の状態

データベース管理システム

TMS

(Truth Maintenance System)

ATMS

(Assumption-Based TMS)

真理性維持システム （TMS）

信念の表現：節点 意味 （正当化）

1．支持リスト正当化（SL）
（SL＜In リスト＞＜Out リスト＞）

2．条件つき正当化（CP）
（CP＜結果＞＜In 仮説リスト＞＜Out リスト＞）

1：矛盾の原因の候補となる信念を求める。
2：矛盾の原因の候補がよくないという意味の節点を生成する。
3：矛盾の原因の候補から適当な信念を選んでOut 状態にして矛盾を解消する。

TMS (Truth Maintenance System)
by Doyle ('79)

信念の他の信念との依存関係を正当化 (justification)

代表的正当化

[1]支持リスト正当化 (SL)
(Support List)
<節点> (SL <In リスト> <Out リスト>
↑ ↑ ↑ 成
立 ← すべて In すべてOut

[2]条件つき正当化 (CP)
(Conditional Proof)
<節点> (CP <結果> <In リスト> <Out リスト>)
↑ ↑ ↑ 成
立 ← すべて In すべてOut

仮説に基づくTMS (ATMS)

Assumption－based TMS by DeKleer ('86)

節点の表現： n： <データ、ラベル、正当化>
データと正当化は、推論システムから供給
ラベルは、節点が成立する<u>環境の集合</u>
ラベルは ATMS が維持 ↑{E_1、E_2、・・・}

ATMSのラベル更新アルゴリズム

1. 節点nの正当化が与えられると、ラベルを求める
2. If 新しいラベル＝もとのラベル、終了
3. If n が矛盾節点、
そのラベルの環境を NOGOOD とし、各節点のラベルから
矛盾する環境を除く Else, n を正当化に含む節点に対して、
このラベルの変化に伴う変更を行う