第１０講 論理による推論（4）
ー述語論理ー

- 述語論理の推論

述語論理の推論

- 述語論理の推論
  - 既存の論理式を組み合わせて、新しい論理式を導くこと
  - はじめに、公理（axiom: 無条件に正しい論理式）を決めておき、推論規則（inference rule）を用いて真となる論理式を導く
  - 公理系 （axiomatic system）
    - 上記の体系

述語論理の公理（論理的公理）

- $P \rightarrow (Q \rightarrow P)$
- $(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))$
- $(P \rightarrow \neg Q) \rightarrow ((Q \rightarrow P) \rightarrow \neg P)$
- $(\forall X) P \rightarrow P_{x/t}$
- $(\forall X) (P \rightarrow Q) \rightarrow ((\forall X) P \rightarrow (\forall X) Q)$
- $P \rightarrow (\forall X) P$

推論規則

- モーダスポネンス（Modus Ponens:MP）
  - 三段論法を一般化
  - ２ちの論理式P, P → Qがともに真であるとき、論理式Qが真になる
    - $$\frac{P \rightarrow Q \quad P}{Q}$$
- 全称化（universal instantiation: UI）
  - 全称∀で束縛された変数Xを任意の項tに置き換えることができる
    - $$\frac{(\forall X)\, P}{P_{X/t}}$$

モーダスポネンスの例

- 父親（太郎、健一） → 男性（次郎）
父親（太郎、健一）
の２つの論理式が真であれば、MPにより
男性（次郎）
真となり、以下の関係が成り立つ。
{父親（太郎、健一） → 男性（次郎），
父親（太郎、健一）} |= 男性（次郎）

論理式の証明（proof）

- $\{P1, P2, \ldots, Pn\} \vdash Q$ の証明
  - $\{P1, P2, \ldots, Pn\}$ 中の論理式や論理的公理からはじめて、これらに推論規則を適用して新しい論理式を作り出していき、最終的にQに至る論理式の系列を求めるプロセス

論理式の証明の説明

- 論理式Σ
  - {母である (花子、太郎) ,
    父である (一郎、花子) ,
    (∀X) (∀Y) (父である (X, Y) → 親である (X, Y)) ,
    (∀X) (∀Y) (母である (X, Y) → 親である (X, Y)) ,
    (∀X) (∀Y) (∀Z) (父である (X, Z) ,
    親である (Z, Y) → 祖父である (X, Y)) }
- 2つの推論規則 (MPとUI) の適用により、
  - 祖父である (一郎、太郎)
  が得られる。
- 上記のような証明が存在することを
  Σ |- 祖父である (一郎、太郎)
  とかく。

融合 (導出) 原理に基づく証明

- 効率のよい証明方法
  - {P1, P2, …., Pn} ∪ {¬Q} が充足不能であることを証明することによって、 {P1, P2, …., Pn} |= Q を証明する
  - 背理法に基づく
  - 背理法とは
    - Pをある命題とすると、Pの否定が偽であることを示すことにより、Pが真であることを導く証明法
    - 例:
      - 命題:すべての人間が動物である
      - 証明:この否定は「少なくても1人の人間は動物でない」である。この命題は明らかに偽である。なぜなら、少なくとも1人の植物人間が存在することになるので、よって、すべての人間は動物である。

融合 (導出) 原理による証明

1. {P1, P2, …, Pn} の個々の論理式を節の集合形式に変換する
2. 証明したい論理式の否定¬Qを節の集合の形式に変換する
3. 2つの節を組み合わせて融合 (導出) 節を作る
4. 空であるような融合 (導出) 節が作られたとき、
   {P1, P2, …, Pn} |= Q
   が証明できたことになる

節 (clause)

- リテラル
  - 正リテラル
    - 素論理式
  - 負リテラル
    - 素論理式の否定
- 節
  - 単一のリテラルまたは、リテラルの選言
  - P1 ∨ P2 ∨ ….. ∨ Pn
  - {P1, P2, …., Pn} とも表現できる
- ホーン節 (Horn clause)
  - 節のうちで、そこの含まれる正リテラルの数が0または1のもの

節集合への変換

- P → Q を ¬P ∨ Q に変換
- 同じ名前の束縛変数が2箇所あるとき、一方の変数名の変換をおこなう
- 否定記号を中に入れる
- 限定記号を外に出す
- 存在記号によって束縛される変数があるとき、その変数を関数表現に置き換えることによって存在記号を消去する。この関数をスコーレム関数という。
- リテラルの連言形式に変換する
- 節形式から節集合に変換する

単一化アルゴリズム

1. 2つのリテラルの不一致集合を求める。不一致集合とは、両者の対応する引数を左から調べて、最初の不一致部分を抜き出したものである。
2. 不一致集合が空であれば、両者は同じリテラルであるので、このアルゴリズムは成功終了する。
3. 不一致集合の各要素において、一方が変数Vで、他方が項Tであれば、V/Tとする。不一致集合の要素で、両方とも非変数なものがあれば、単一化は不可能なので、このアルゴリズムは失敗する。
4. P {V/T} とQ {V/T} を新しいPとQとし、ステップ1へ

融合（導出）節の生成

1. 節集合の中から2つの節CとDを選択する
2. CとDの融合（導出）節Rを作る。融合（導出）節とは、節Cから正リテラルL, Dの中から負リテラル¬L'を選び、LとL'が適当な代入θによって単一化できるとき、CθからLθを除いた節C'とDθから¬L'θを除いた節D'の和集合C'∪D'である。
3. 融合（導出）節に節集合を加えて、1から3までのステップを繰り返す。