論理と計算

第６回
述語論理：構文・意味・解釈

担当：尾崎　知伸
ozaki.tomonobu@nihon-u.ac.jp

講義予定　※一部変更（前倒し）になる可能性があります

09/22	01. オリエンテーション と 論理を用いた問題解決の概要
09/29	02. 命題論理：構文・意味・解釈
10/06	03. 命題論理：推論
10/13	04. 命題論理：充足可能性問題
10/20	05. 命題論理：振り返りと演習（課題学習）
10/27	06. 述語論理：構文・意味・解釈
11/03	07. 述語論理：推論　※文化の日，文理学部授業日
11/10	08. 述語論理：論理プログラムの基礎
11/17	09. 述語論理：論理プログラムの発展
11/24	10. 述語論理：振り返りと演習（課題学習）
12/01	11. 高次推論：発想推論
12/08	12. 高次推論：帰納推論の基礎
12/15	13. 高次推論：帰納推論の発展
12/22	14. 高次推論：振り返りと演習（課題学習）
01/19	15. まとめと発展的話題

目次：今回の授業の内容

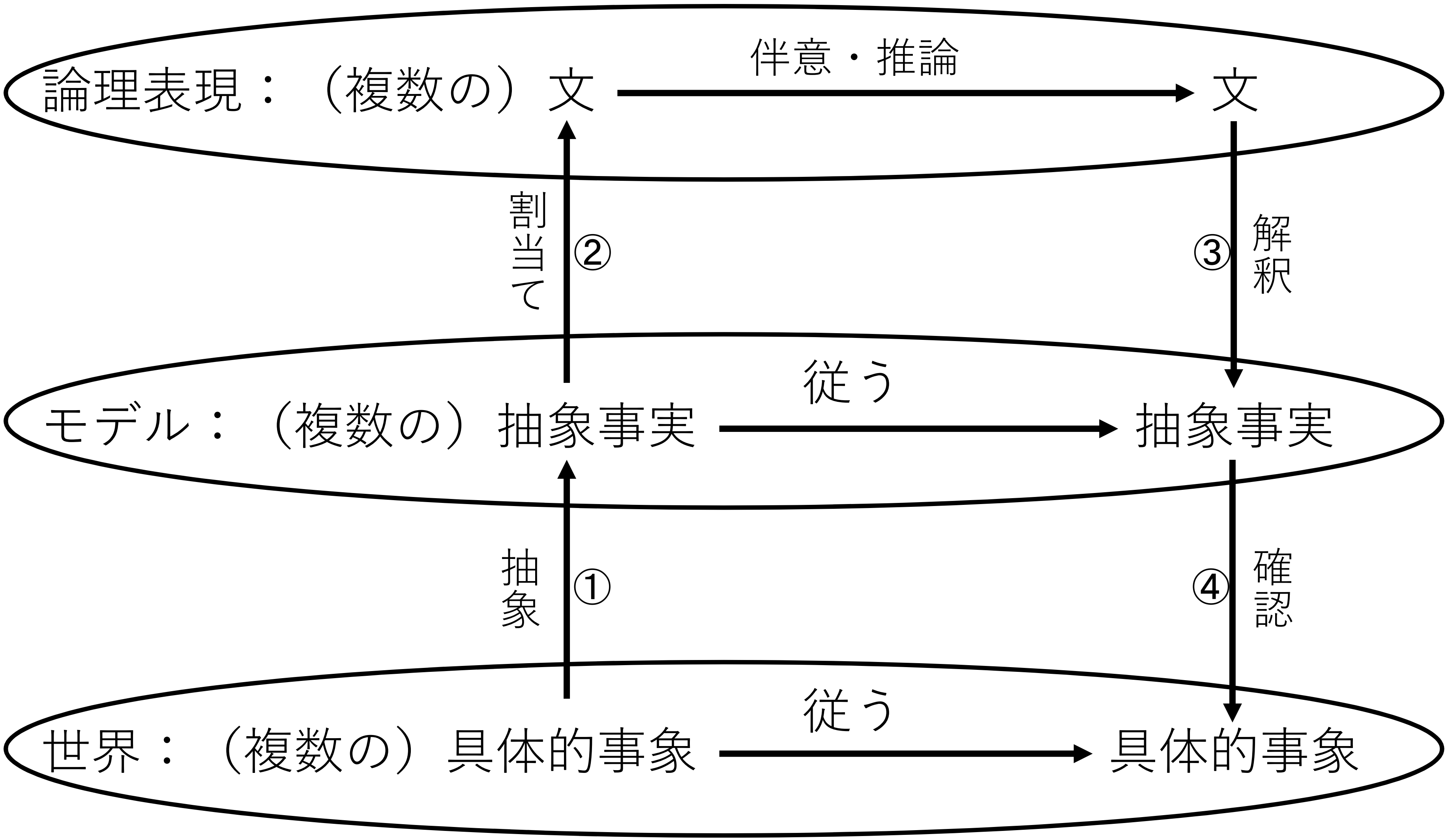
- 復習
  - 論理学における世界の表現
  - 命題論理
- 述語論理
  - 構文
  - 解釈
  - 分類

これまでの資料より

論理学による世界の表現

論理学による世界の表現

- 命題論理や述語論理で物事を表現する目的
  - 真偽についての一般的法則性を利用して「問題を解決する」こと



論理による世界の表現とモデル

- 問題解決の手順
  - ※難しく感じるかもしれませんが，「プログラミング」や「データ分析」と同じです
  - 「解きたい問題をプログラミング言語で表記し，実行することで答えを得る」
  - 「対象に関する属性間の関係を抽象化（簡略化）したモデル（数式）を考える」
- 手順１：世界の抽象表現を与える
  - 実世界は多様な構成要素と関係から成り立っている
  - 問題解決に関連する部分のみを切り出す（取り出す）
    - 細部は無視して，特定の属性に関する真偽のみを問題にする⇒抽象化
  - モデル：世界の抽象表現に対応する
    - モデルの表現は論理体系で異なる
- 手順２：モデル要素に対する割り当て（assignment）
  - モデルの各要素に対して，命題記号や述語記号を対応させる
  - 対応後は，論理体系の持つ推論能力（推論規則）を用いて，新たな文を導出する
- 手順３：割り当ての逆操作（解釈）
  - 文に対し，それがモデルにおいて何を意味するかを与える
- 手順４：確認
  - 抽象事実に対する具体事実が，本当に成り立っているかを確認する

モデル・記号化・解釈

- モデル
  - 抽象化：細部は無視し，特定の属性に関する真偽のみを問題とする
  - 与えられた世界（=対象領域）の抽象表現を，その世界のモデルと呼ぶ
  - （例）自動車のモデル：「色が白い」，「大きさが大きい」，「車種は救急車」，「サイレンが有る」など，必要と思われる属性のみを表現する．
  - メーカ，所有者，走行距離など，必要ないと思われるものは表現しない（抽象化）

- 記号化（representation）
  - モデルの各要素に対して命題記号（あるいは述語記号）及び対象記号を対応させる
  - モデルの各要素を記号（命題記号・述語記号）で表現できれば，論理体系の持つ推論能力を用いて，事実の集合から論理的に導かれる新たな事実を知ることができる
  - （例）自動車のモデル：p ≡ 色が白い，q ≡大きさが大きい，r ≡車種は救急車， s ≡ サイレンが有る

- 解釈（interpretation）
  - 記号化の逆
  - （推論結果である）記号（命題記号・述語論理）の文が与えられたときに，それがモデルにおいて何を意味するかを与える
  - （例）自動車のモデル：$p \land q$ ≡ 色が白くて大きい， $p \land q \Rightarrow r$ ≡色が白くて大きければ車種は救急車

論理学の三層構造の意味

- 論理学は対象の構造，対象間の関係を問題にする
  - 論理は，対象となる世界（＝対象領域）を表現するための形式言語＝表現言語
  - 記号が「何を表すのか」，「何を意味するのか」ということを考えずに，「記号操作のみで論理的な操作を行う」ことができる

  - 論理は各文の本当の意味を理解していない．それらの真偽だけに注目
  - 得られた論理的帰結に対して，p,qなどの記号が，その世界で何を表しているかの「解釈」を与えると，それは意味を持つ
  - 言葉の意味は理解できないが，正しい推論はできる
    - 抽象表現系での推論は，具体的な意味に依存しない

- 三層構造の意味
  - 抽象化は，論理学の扱う範囲を文の真理値に限る役割を果たしている
  - 記号化は，論理学がモデル依存になることを防いでいる
  - 記号化によって，異なる問題が同じ記号表現を取ることを妨げない
    - 問題によらない，普遍的な表現と推論方法を提供している

命題論理

命題論理（propositional logic）

- 命題を基本構成要素とする論理言語
- 命題：真または偽という性質をもつもの．真偽の判断の対象となる文章や式
  - 命題という用語は，高校数学でも出てきたかと思います
  - 「真偽を問題にすることができる」ことが特徴
    - 逆に言えば，真偽を問題にすることができないものは命題とはならない
      - 命令文・感嘆文・挨拶などは，命題にならない

- 命題論理の構成要素
  - 文（sentence）＝命題論理式（文と式を同じ意味で使います）
  - 原子文（atomic formula）：最も単純な文（それ以上分解することのできない命題）
    - p：「患者は微熱がある」，q：「患者は咳をする」，
      r：「患者は疲れやすい」，s：「患者の病気＝肺結核」
  - 複合文（complex sentence）：文を結合子（connective）でつなげた文
    - 結合子（connective）：連言（conjunction∧），選言（disjunction∨），
      否定（negation¬），含意（implication⇒），同値（equivalence⇔）
    - $p \land q \land r \Rightarrow s$
      「患者が微熱があり，咳をして，疲れやすい ならば 患者の病気=肺結核」
  - 一つの特定の対象に対する記述

命題文の構文

- 命題論理の出てくる記号：論理定数・命題記号・結合子・括弧
- 命題文：命題論理に出てくる論理式（命題記号の組み合わせ）
  - （命題文＝命題論理式：両者を同じ意味で利用します）
- 命題文は原子文（atomic formula）と複合文（complex formula）からなる
- 命題文を結合子（connective）で繋いだものを複合文と呼ぶ
  - 結合子には，否定記号・連言記号・選言記号・含意記号・同値記号が含まれる

[命題文の（言葉による）定義]

1. 論理定数 true と falseは文である
2. 命題記号 p, q, r … は文である（命題記号＝命題変数）
3. 文を括弧で囲ったものは文である
4. 文の先頭に否定記号¬を付与したものは文である
5. 二つの文を連言記号∧で繋げたものは文である
6. 二つの文を選言記号∨で繋げたものは文である
7. 二つの文を含意記号⇒で繋げたものは文である
8. 二つの文を同値記号⇔で繋げたものは文である
9. 1.〜8. で作られるもののみが文である

- 再帰的な定義：「命題文」の定義に「命題文」が出現しています

命題文	$\rightarrow$	原子文 | 複合文
原子文	$\rightarrow$	true | false | p | q | r | ...
複合文	$\rightarrow$	（命題文）
	|	¬ 命題文
	|	命題文 ∧ 命題文
	|	命題文 ∨ 命題文
	|	命題文 ⇒ 命題文
	|	命題文 ⇔ 命題文

結合力（降順）：¬, ∧, ∨, ⇒, ⇔

命題論理の意味（Semantics）

- 命題文は，世界の真偽のみに着目して抽象化を行っている
- 真偽のみが意味を持つ

[命題文の意味の定義]

1. 文true は真を意味する．文falseは偽を意味する
2. 命題記号（p,q,r,⋯）は真偽のいずれかの意味を取る
3. 複合文の意味は真理値表（truth table）によって定める

プログラミングの「条件式」の考え方に近い

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

- ブール代数と同じ
- 真理値は真偽値とも呼ばれる
- 複合文の真理値は，文を構成する各命題文の真理値計算を（再帰的に）繰り返すことで獲得される

命題論理の意味と現実世界における直観的な意味

- 否定文：pではない
- 連言文：pかつq
- 選言文：pもしくはq （p, qの少なくとも一方が）

- 含意文：pならばq
  - 含意文p⇒qの意味に注意
  - 前提pがfalseのときは，帰結qの真偽に関わらず（式として）trueとなる
  - 一般に，前提が成り立たないときには規則は適用されないが，規則としては間違ってはないので「true」とするのが妥当である
  - 命題論理では，p,q間に因果関係や関連性が要求されない
    - “5が奇数なら，日本の首都は東京である”という文は，（前提，帰結間に意味的な繋がりがないという点で）現実的には明らかにおかしい
    - しかし，命題論理では（前提も帰結もtrueなので）trueとなる

- 同値文：pとqは同値
  - *p* は *q* が正しい（trueである）とき，またそのときに限り正しい（trueである）
  - 英文だと，”*p* if and only if *q*” や”*p* iff *q*”と表記する

命題論理における解釈（interpretation）とモデル（model）

［解釈の定義］

- 命題文αに現れる原子文（命題記号）を$\alpha_1, \alpha_2 \cdots \alpha_n$とする．
- 各原子文α_iへの真理値(true,false)の割り当てをαの解釈と呼ぶ

ここでの［解釈］は
各原子文への真理値割り当て
を表す「技術用語」です

- それぞれの「割り当ての仕方」を解釈と呼ぶ
  - n個の原子文があるときは，2^n個の解釈が存在する
  - 例： p∧q∨r の解釈

{p = true, q = true, r = true}, {p = true, q = true, r = false}, {p = false, q = true, r = true}, {p = false, q = true, r = false },
{p = true, q = false, r = true}, {p = true, q = false, r = false}, {p = false, q = false, r = true}, {p = false, q = false, r = false }

- 略記法：trueが割り当てられた命題記号からなる集合で表現する
  - { p, q, r }, { p, q }, { q, r }, { q }, { p, r }, { p }, { r }, { }

［モデルの定義］

- 命題文αの真理値を真とする解釈Iをαのモデルと呼ぶ.

解釈やモデルは，真理値割り当ての集合
モデルの全体集合は，
解釈の全体集合の部分集合 となる

- 真理値表
  - 各行がそれぞれ一つの解釈
  - 命題文が真となる行がモデル

p	q	$\neg p \vee q$
$false$	$false$	$true$
$false$	$true$	$true$
$true$	$false$	$false$
$true$	$true$	$true$

文 ¬p∨qのモデルは
{ }, { q }, { p, q }

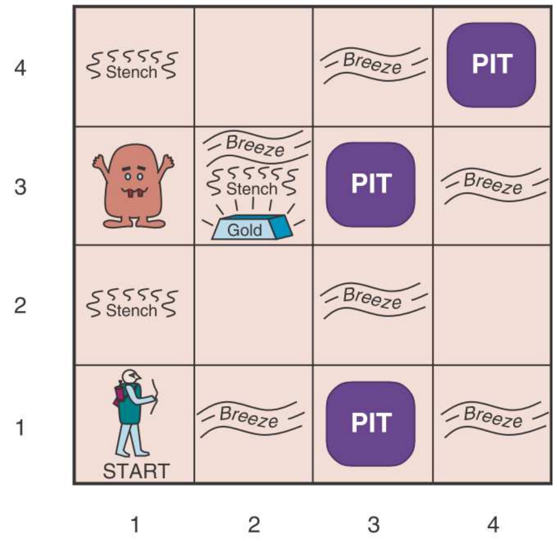
文中の各命題変数に対する真理値割り当てと，それに対する結果を表にしたもの

Wumpus World

- 通路で結ばれた4x4の部屋からなる洞窟の世界
  - 「洞窟」内には，獣（wumpus）x 1と人間（知識エージェント）がいる
  - 獣（wumpus）は，自分の部屋に入ってきたエージェントを食べる
  - 人間は，矢を用いて，獣を殺すことができる（矢は一本しか持っていない）
  - 幾つかの部屋には大きな穴があり，迷い込んだ人間は，穴に落ちて死んでしまう
  - ただし，獣は大きいので穴には落ちない
  - 人間の目的は，黄金の山を見つけること

- 知覚情報は5種類（「洞窟」ということで，人間の知覚情報は限定的である）
  1. 獣がいる部屋とその隣（上下左右）の部屋で，悪臭（stench）を感知する
  2. 穴が開いている部屋の隣の部屋で風（breeze）を感知する
  3. 黄金が置いてある部屋で輝き（glitter）を感知する
  4. 壁にぶつかると衝撃（bump）を感知する
  5. 何処にいても，獣が人間に殺されたときに発する叫び声（scream）を感知する

⇒受け取る知覚を5つの記号からなるリストで表現する
悪臭と風を感じた場合⇒[Stench, Breeze, None, None,
一方，場所は [x, y]と表現 する



行動をしながら
新たに分かったことを随時
知識ベースに追加していく

Wumpus Worldにおける知識ベース

- 部屋の穴のみを対象に，Wumpus Worldの知識ベースを構築してみよう
  - 今回は大変なので，[1,1] → [2,1]への移動までを対象とする

- 命題記号
  - 各i, jについて，$P_{i,j}$は部屋[i,j]に穴がある　　を意味する命題変数
  - 各i, jについて，$B_{i,j}$は部屋[i,j]で風を感じる　を意味する命題変数

- 知識ベース
  - 部屋[1,1]には穴がない　R1：$\neg P_{1,1}$
  - 隣の部屋に穴があるとき（またそのときに限り）風を感じる
    - R2：$B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})$
    - R3：$B_{2,1} \Leftrightarrow (P_{1,1} \lor P_{2,2} \lor P_{3,1})$　　※本来はすべての部屋について準備する
  - 部屋[1,1]と部屋[2,1]での知覚
    - [1,1]で風を感じない　R4：$\neg B_{1,1}$
    - [2,1]で風を感じた　　R5：$B_{2,1}$
  - 知識ベースは，$R1 \land R2 \land R3 \land R4 \land R5$となる
    - 知識ベースに対するすべてのモデル（知識ベースを真にする割り当て）において，$P_{i,j}$の真偽値を確認すれば穴がある・ない・判断不能が分かる

真理値表を書くことで
知りたい事柄（命題式）の
真偽を判定する

述語論理：構文

命題論理から一階述語論理へ

- 命題論理（propositional logic）
  - 「命題」を基本要素とする論理言語 → 命題のみしか扱うことができない
  - 表現能力に乏しい場合も．．
    - 多くのオブジェクトが存在する場合に，それを簡潔に記述することができない
      - 例：「穴のある部屋の隣では風を感じる」in Wumpus World
        ルールは一つだが，実際には，各部屋についてそれぞれ記述する必要がある．
    - オブジェクト間の関係を簡潔に記述することができない
      - 例：親子関係，各親子関係についてバラバラに命題を準備しなければならない

- 命題論理の基本要素（文脈非依存，無曖昧性，宣言的など）を土台に表現力を豊かにする
  - 自然言語における統語論から．．．．
  - 「オブジェクト」(object)に相当する名詞・名詞句（部屋，穴など）
  - 「オブジェクト間の関係」（relation）に相当する動詞・動詞句（隣接している）
    - 関係の一種としての単項関係＝性質
    - 関係の一種としての「関数」(function)：入力に対して唯一の値が存在する関係

- 述語論理（predicate logic）
  - 述語論理：「オブジェクト」と「関係」を基本要素とする論理言語
  - 一階述語論理（first-order logic）：オブジェクトに対する量化（変数化）を許す
  - 二階述語論理（second-order logic）：述語や関数記号に対する変数化を許す
    - 高階述語論理（higher-order logic）

述語論理（predicate logic）

- 述語論理の特徴
  - 世界に登場する「個々の対象」を表現できる
  - 対象と対象の関係を中心に考え，個々の対象間の「関係」を述語を用いて表現する
  - 無限の対象からなる世界を表現できる
  - 準決定的である（正しい文は反駁証明可能．正しくない文は，有限時間で証明できるとは限らない）
- 述語論理の構成要素
  - 述語文・述語論理式
  - 対象と述語（対象の性質や対象間の関係）の2種
    - 対象：文中に現れる主語や目的語に相当 / 変数にしても良い
    - 述語：動詞や形容詞に相当
      - "述語（対象，対象．．．．）"の形式で記述します．
      - orbits(earth, sun) / parent(tom, jack) / fever(ann)
      - catch_a_cold(X)⇒have_a_cough(X)
  - 命題論理と同様，結合子を用いて複合文を構成する
  - 限量子（quantifier）
    - 全称限量子（universal quantifier）∀：「すべてのXに対して～」
    - 存在限量子（existential quantifier）∃：「あるXに対して～」
    - ∀X ∃Y child(X, Y)

述語論理式の例

※読みやすさのため
∀,∃の範囲に( )ではなく [ ] を利用しています

- ∃X [have(I, X) ∧ computer(X)] : 私はコンピュータを持っている
- ∀X [member(X, is) ⇒know(X,c_lang)]：情報科学科の学生はC言語を知っている
- ∀X [bird(X) ∧ ¬penguin(X) ⇒ fly(X)]：ペンギン以外の鳥は飛ぶ
- ∀X [animal(X) ⇒ has(X, parent(X))]： すべての動物は生みの親を持っている
- ∀X [human(X) ⇒ ¬connect(chin(X), elbow(X)]：すべての人間は，顎と肘をつけられない
- ¬∃X [human(X) ∧ connect(chin(X),elbow(X)]：顎と肘がつく人間はいない
- ∀X, Y, ∃P [parent(X, P) ∧ parent(P, Y) ⇒grandparent(X,Y)] ：親の親は祖父母である
- ∀X, Y [student(X) ∧ lecture(Y) ⇒understand(X, Y)] : 学生はすべての授業を理解できる

∀,∃を無視すれば読める!

( )を伴う文字列：関係を表す述語
大文字で始まる文字列：対象を表す変数
小文字で始まる文字列：対象を表す定数
∧（かつ），¬（否定），⇒（ならば）

一階述語論理の構文（Syntax）

命題論理との違いに着目しよう

文	$\rightarrow$	原子文 $\mid$ 複合文
原子文	$\rightarrow$	述語記号 $\mid$ 述語記号 (項,$\cdots$) $\mid$ 項 $=$ 項
複合文	$\rightarrow$	（文）
	$\mid$	$\neg$ 文
	$\mid$	文 $\wedge$ 文
	$\mid$	文 $\vee$ 文
	$\mid$	文 $\Rightarrow$ 文
	$\mid$	文 $\Leftrightarrow$ 文
	$\mid$	限量子 変数, $\cdots$ 文
項	$\rightarrow$	定数
	$\mid$	変数
	$\mid$	関数記号 (項,$\cdots$)
限量子	$\rightarrow$	$\forall \mid \exists$
変数	$\rightarrow$	大文字で始まる文字列
定数	$\rightarrow$	小文字で始まる文字列 $\mid$ 数値
関数記号	$\rightarrow$	小文字で始まる文字列
述語記号	$\rightarrow$	true $\mid$ false $\mid$ 小文字で始まる文字列

結合力（降順）：$\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

項（Term）：オブジェクトの論理表現

変数（Variable）：オブジェクトの抽象表現

限量子（quantifier）：量化（変数化）

項（Term）

- 項：オブジェクトの論理表現
  - 問題領域に依存して決められる
  - 述語の引数に用いられる

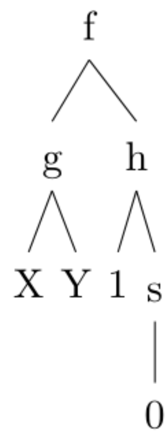
- 項の定義
  - 定数（constant）は，項である
  - 変数（variable）は，項である
  - fをn引数の関数記号（function symbol）と，t_1, … , t_n を項としたとき，f(t_1,…,t_n)は項である
  - 以上で定義されるものだけが項である

- 変数・定数は，それぞれ与えられた集合に含まれる要素である必要がある
  - 集合を明示しないことも多い → この授業では，定数・変数を以下のように表現する
    - 書籍や資料によっては，大文字・小文字が逆になっているケースもあります．
  - 定数：小文字で始まる文字列or数値
  - 変数：大文字で始まる文字列

関数記号（function symbol）と複合項・基礎項

- 項の定義：再帰的な定義になっていることに注意

  → 関数記号を用いた入れ子構造（複合項）を許す
  - 例 : f(g(X, Y) , h(1, s(0)))
  - 項は，関数記号を中間ノード，引数を枝，定数・変数を葉とする木



- 複合項（compound term）：関数記号を伴う項

- 関数記号(function symbol)：構造を持った複雑なオブジェクトを表現するための手段
  - 述語論理における関数記号は，代数におけるそれとは異なる
  - 代数における関数記号：明示的な計算結果（計算される値）を考える
  - 論理における関数記号：単に記号パターンのみが意味を持つ
    - 例：pc1のメモリ
      - memory(pc1) .. 代数では，計算結果(例えば，4GBなど)を考える
      - memory(pc1) .. 論理では，単なる木構造．「計算」手続きは考えない

- 定数（constant）：0引数関数に相当

一階述語論理では，
関数記号の変数化はNG
ex. f(X(a, b) , c) はNG

- 基礎項（ground term）：変数を全く含まない項．
  - 基礎項は，問題領域中に現れる具体的な対象を表す
  - 一方，変数を含む項は，（ある種の）パターンを表す．ex : memory(PC)

文（sentence）

- 文（述語論理文）は，原子文と複合文に分類できる

- 原子文（atom, atomic sentence）の定義：
  - p をn 引数の述語記号とし，$t_1, \cdots, t_n$を項とする．

    このとき $p(t_1,\cdots,t_n)$を原子文と呼ぶ（なお$n = 0$のときは括弧を省略し p と書く）

  ※見た目は複合項と似ていますが，原子文と複合項はまったく異なるものです

- 原子文：「オブジェクト間の関係」「オブジェクトの性質」の論理表現
  - オブジェクトに対する事実に相当する
  - 0引数述語：命題論理における原子文（単一の命題記号からなる文）と同じ
  - 1引数述語：オブジェクトの性質を表すことが多い
  - n引数（$n > 1$）述語：オブジェクト間の関係を表すことが多い

- 基礎原子文（ground atom）：
  - すべての引数が基礎項である原子文（変数を含まない原子文）

    真偽値を割り当てる対象

- 複合文（complex sentence）：
  - 論理結合子（logical connective）を用いて原子文を連結した文（意味論は命題論理と同じ）

限量子（quantifier）

- 限量化（量化）：論理式が適用される議論領域の個体の「量」を限定すること
  - 実際には，「変数」に対して量を考えることに相当する
  - 一階述語論理における限量は「全称」と「存在」の2種類

- 全称限量（universal quantifier）：すべてのオブジェクトに対する記述
  - $\forall X\,(LS)$：　「すべてのオブジェクトxについて論理式（文）LSは真である」
  - 例：$\forall X\,(\mathrm{mammal}(X) \Rightarrow \mathrm{animal}(X))$
    - 「すべての哺乳類は動物である」（xが哺乳類であれば，それは動物である）
    - 対象（オブジェクト）が { dog, cat, eagle} の場合，　以下の論理式と同じ
    $(\mathrm{mammal}(dog) \Rightarrow \mathrm{animal}(dog)) \wedge (\mathrm{mammal}(cat) \Rightarrow \mathrm{animal}(cat)) \wedge (\mathrm{mammal}(eagle) \Rightarrow \mathrm{animal}(eagle))$
    - $\forall X\,(\mathrm{mammal}(X) \wedge \mathrm{animal}(X))$ との違いに注意！

- 存在限量（existential quantifier）：あるオブジェクトに対する記述
  - $\exists X\,(LS)$：　「少なくとも一つのオブジェクトxについて論理式（文）LSは真である」
  - 例：$\exists X\,(\mathrm{mammal}(X) \wedge \mathrm{has\_beak}(X))$
    - 「哺乳類の中には，くちばしを持つものがいる」（カモノハシですね．．．）
    - 対象（オブジェクト）が { dog, platypus} の場合，　以下の論理式と同じ
    $(\mathrm{mammal}(dog) \wedge \mathrm{has\_beak}(dog)) \vee (\mathrm{mammal}(platypus) \wedge \mathrm{has\_beak}(platypus))$

限量子の入れ子とスコープ

曖昧性がない場合は，
∀X∀Yを∀X,Y,
∃X∃Yを∃X,Y
と表現することもある

- 複数の限量子を用いることで，より複雑な文を表現する
  - 例：「親子関係は祖先関係である」 ∀X ∀Y (parent(X, Y)⇒ancestor(X, Y))
  - 例：「すべての動物は心臓を持つ」 ∀X ∃Y (animals(X)⇒ has(X, Y), heart(Y))
- 順序は非常に大切 （括弧を付けて考えよう）
  - ∀X ∃Y loves(X, Y) ：「誰もが誰かを愛している」
    - ∀X(∃Y (loves(X, Y)))
    - Everybody loves somebody : すべての人について，それぞれが誰か愛する人がいる
  - ∃Y ∀X loves(X, Y) ：「すべての人に愛される（ある）人が存在する」
    - ∃Y(∀X (loves(X, Y)))
    - There is someone who is loved by everyone：ある一人の人がいて，その人は全員から愛される
- 限量子の有効範囲（スコープ）：括弧の内側
- 束縛変数（bound variable）：スコープ内にある変数・限量化されている変数
- 自由変数（free variable）：スコープ外にある変数・限量化されていない変数
- 閉論理式（閉じた文）：すべての変数が束縛されている述語文
- 開論理式（開いた文）：限量化されていない変数を含む文（真偽は定まらない）
  - 変数に対して世界の適当な対象を割り当てて，その述語文が成り立つような解釈を求めるような場合に必要となる（この授業では対象外）

∀と∃の関係

- 以下の文は同じ内容を表す
  - 　∀X ¬dislikes(X, programming)：すべてのxは，programmingが嫌いではない
  - ¬∃X　dislikes(X, programming)：programmingが嫌いなxは存在しない
- 以下の文は同じ内容を表す
  - 　∀X　likes(X, logic)：すべてのxは，logicが好き
  - ¬∃X ¬likes(X, logic)：logicが好きではない人は存在しない

- ∀は連言（∧），∃は選言（∨）
  - ∧と∨に関するド・モルガンの法則が適用可能
- ド・モルガンの法則（¬の移動 と ∀⇔∃の交換）
  - ∀X (¬LS) ⇔ ¬∃X (LS)
  - ∀X (LS) ⇔ ¬∃X (¬LS)
  - ¬∀X (LS) ⇔ ∃X (¬ LS)
  - ¬∀X(¬LS) ⇔ ∃X (LS)

- ∀と∃は片方あれば十分か？
  - 表現としては十分
  - 可読性のために両方を用いるのが一般的
    - 結合子がすべて必要なわけではないのと同じ

「すべての人間は死ぬ運命にある」
　∀X (human(X)⇒mortal(X))　⇔
¬∃X¬ (human(X)⇒mortal(X)) ⇔
¬∃X¬ (¬human(X)∨mortal(X)) ⇔
¬∃X　(¬¬human(X)∧¬mortal(X)) ⇔
¬∃X　(human(X)∧¬mortal(X))
「人間で死ぬ運命にない人はいない」

等号関係

- 等号（equality symbol）：
  - 二つの項が同じオブジェクトを参照するという意味の記述
  - 原子文を作るもう一つの方法
  - 例：father(john) = henry

  - ※後学する単一化（Unification）とは別の概念です
    - 意図としては，両者が等しいということですが，
      単純に「2引数の述語記号= に対する中記法」と捉える方が分かりやすいと思います

- 否定との組み合わせ
  - 二つの項が同一のオブジェクトではないことを主張
  - 例：∃X∃Y brother(john, X)∧brother(john, Y)∧¬(X=Y)
    - Johnには，少なくとも2人の兄弟がいる．

    - 注意：∃X∃Y brother(john, X)∧brother(john, Y)　は，XとYが等しい場合も成立する
      - 兄弟が一人の場合でも真となってしまう．

述語論理：利用例

一階述語論理の利用例

- 親族関係
  - オブジェクト（項）：人（今回は変数）
  - 関数記号：父（father），母（mother）
  - 関係（述語）：親（parent），子（child），母親（mother），性別（male,female）

  - $\forall X\ (\neg \mathrm{male}(X) \Leftrightarrow \mathrm{female}(X))$　男性と女性は排他的なカテゴリである
  - $\forall P, C\ (\mathrm{parent}(P, C) \Leftrightarrow \mathrm{child}\ (C, P))$　親と子は互いに逆関係である
  - $\forall M, C\ (\mathrm{mother}(C) = M \Leftrightarrow \mathrm{parent}(M, C) \wedge \mathrm{female}(M))$
    - ある人（C)の母（M）は，その人の女親である．
  - $\forall G, C\ (\mathrm{grandparent}(G, C) \Leftrightarrow \exists p\ \mathrm{parent}(G, P) \wedge \mathrm{parent}(P, C))$
    - ある人の祖父母は，その人の親の親である

- aさんの父方の祖先
  - オブジェクト（項）：a さん
  - 関数記号：父（father）
  - 関係：父方の祖先（fanc）

  - $\mathrm{fanc}(\mathrm{father}(a))$　aさんの父親は，aさんの父方の祖先
  - $\forall X\ (\mathrm{fanc}(X) \Rightarrow \mathrm{fanc}(\mathrm{father}(X)))$
    - Xがaさんの父方の祖先ならば，Xの父はaさんの父方の祖先

一階述語論理の利用例

- 自然数
  - 定数記号：0
  - 関数記号：s (後続関数, successor function：sの次の値)
  - 関係：自然数（natNumber）

s(0), s(s(0)), s(s(s(0))), …

natNumber(s(0))　　1 (0の次の値)は自然数である

∀N (natNumber(N) ⇒ natNumber(s(N)) 　　Nが自然数であれば，N+1は自然数である

自然数の加算
0 + N = N
N+ M = P ⇒ (N+1) + M = (P+1)

∀N (natNumber(N) ⇒ plus(0, N, N))
∀N, M,P (natNumber(N) ∧ natNumber(M) ∧ natNumber(P) ∧
plus(N, M, P) ⇒ plus(s(N), M, s(P)))

減算・乗算も考えてみよう

意味解釈の多様性

山手線の駅：隣の駅
偶数：2を足す
天皇？：長男？

- aさんの父方の祖先
  - fanc(father(a))
  - ∀X (fanc(X) ⇒ fanc(father(X))

- 自然数
  - natNumber(s(0))
  - ∀N (natNumber(N) ⇒ natNumber(s(N))

- 両者はまったく同じ形
  - pred(func(c))
  - ∀A (pred(A) ⇒ pred(func(A))

  - 各記号が何を意図するのかは人間が判断しているだけで，機械には関係ない．
  - 利用者（人間）が実世界との対応関係を与えているに過ぎない．

述語論理：解釈とモデル

意味と解釈

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \Rightarrow q$	$p \Leftrightarrow q$
$false$	$false$	$true$	$false$	$false$	$true$	$true$
$false$	$true$	$true$	$false$	$true$	$true$	$false$
$true$	$false$	$false$	$false$	$true$	$false$	$false$
$true$	$true$	$false$	$true$	$true$	$true$	$true$

- 述語論理の意味：（命題論理同様）真理値

- 文の真理値は，命題論理と同じように決めればよい
  - 実際には，「基礎原子文」に対する真理値を考えることになる
  - 変数は，領域（後述）を使って具体化する
  - 限量子に関しては，∀は連言に，∃は選言に展開して考えればよい

- 述語論理の解釈：
  - 解釈 .. 対象に対する真偽値の割り当て
  - 述語論理では，以下の二段階で考える
    - 文に現れる定数および関数に対する世界の中での対象物との対応関係を決める
    - 基礎原子文に対する真理値を決める

  - 詳細は，次ページ

解釈

命題論理に帰着
基礎原子文＝命題

- 領域：述語文αが表現する世界での対象の集合

- 解釈：述語文αの領域をDとする．このとき，αの解釈を以下のステップで与える
  1. α中の定数記号，関数記号への割り当て
     - 定数記号c にDの元c^Iを割り当てる
     - n引数の関数記号fにD上の関数：f^I:D^n→Dを割り当てる
  2. 基礎原子文への真理値の割り当て
     - 上記の割り当てにより，α中のすべての述語に対して，それらの基礎原子文がすべて決まる．それらの各基礎原子文に対して，真理値｛true, false｝を割り当てる
     - すなわち，n引数の述語記号pにD上の関数：p^I:D^n→{true, false} を割り当てる

- モデル：αを真にする解釈Iを，αのモデルと呼ぶ
  - 文αの真理値の決定
    - 基礎原子文の真理値は，解釈で直接与えられる
    - 複合文の真理値は，結合子（¬，∧，∨，⇒，⇔）の真理値表から計算
    - ∀X (P)の真理値は，D中のすべての要素xについてPがtrueならばtrue
      そうでなければfalse
    - ∃X (P)の真理値 は，D中の少なくとも一つの要素xについてPがtrueならばtrue
      そうでなければfalse

- 例：ジャンケン（rock-paper-scissors）
  - 文∀X∃Y (Y = lose(X) ⇔ win(Y, X))
  - 領域D = { グー, チョキ, パー }
  - 定数記号：r(ock), p(aper), s(cissors)
  - 関数記号： lose(X) .. Xが負ける手（Xに勝つ手）
  - 述語記号：win(X, Y) .. XはYに勝つ

r → グー
s→チョキ
p→パー
lose(r)→パー
lose(s)→グー
lose(p)→チョキ

win(r,r)→false
win(r,p)→false
win(r,s)→true

win(p,r)→true
win(p,p)→false
win(p,s)→false

win(s,r)→false
win(s,p)→true
win(s,s)→false

win(lose(p), p)→true
win(lose(p),lose(r))→true
..
win(lose(lose(p)), p)→false

関数記号を考えると，
無限個の基礎原子式が
考えられてしまいます．．

- 例：述語文G =nat(a) ∧ ∀X(nat(X)⇒nat(s(X))) ∧ ∀Y(nat(Y)⇒p(Y,s(Y)))
- 領域D：0を含む自然数 （領域は無限集合でもOK）
- 解釈：
  - 定数記号aに，Dの元0を割り当てる
  - 1引数の関数記号sに，$s^I(i) = i + 1$ を割り当てる
  - 述語記号natに関して nat(a), nat(s(a)), nat(s(s(a))), …, にtrueを割り当てる
  - 述語記号pに関して p(a, s(a)), p(s(a),s(s(a)), … にtrueを割り当てる

  - p(X,Y) の意図を 「X < Y」 とすれば，この解釈はモデルとなっている
  - p(X,Y) の意図を 「X以上の最小の素数はY」とすれば，この解釈はモデルとならない

述語論理：分類

述語論理文の分類と論理的帰結

- 命題論理と同様，3種類に分類
  - 恒真：すべての解釈のもとで真となる文
  - 恒偽：すべての解釈のもとで偽となる文
    - 矛盾（contradiction），充足不能（unsatisfiable）とも呼ぶ
  - 充足可能：文が真となる解釈が存在する

- 論理定数true は恒真である
- 論理定数falseは恒偽である.
- 任意の基礎原子文αは充足可能である

- 定理：$G = \{\gamma_1, \gamma_2, \ldots\}$を述語文の集合，$\alpha$を述語文とする．伴意式$G \vDash \alpha$が成り立つのは，$(\gamma_1 \wedge \gamma_2 \wedge \cdots) \Longrightarrow \alpha$が恒真のとき，かつ，そのときのみである
  - $(\gamma_1 \wedge \gamma_2 \wedge \cdots) \Longrightarrow \alpha$が恒真 $\Leftrightarrow$ $(\gamma_1 \wedge \gamma_2 \wedge \cdots \neg \alpha)$が恒偽

- 恒真（恒偽）のチェックは，命題論理の場合以上に困難
  - すべての世界，すべての解釈について調べることは困難 → 証明論が必要
    - ※対象を限定し，陽に解釈・モデルを考えるのもありかとは思います

解釈による文の分類

- 文$\alpha = \forall X \exists Y$ (love (X, Y)) が，恒真，恒偽，充足可能のいずれかを判定したい．
  - 領域 D = { 学生A, 学生B }
  - 定数記号：a, b　　　　　対応関係：a→学生A, b→学生B
  - 述語記号：loves(X, Y) ... XはYを愛している
  - 基礎原子文：loves(a,a)，loves(a,b)，loves(b,a)，loves(b,b)
  - α より $\underbrace{(\text{loves}(a, a) \vee \text{loves}(a, b))}_{\exists} \underset{\forall}{\wedge} \underbrace{(\text{loves}(b, a) \vee \text{loves}(b, b))}_{\exists}$ の真理値を考える
- 同様に，文$\beta = \exists X \forall Y$ (loves(X, Y))恒真，恒偽，充足可能のいずれかを判定したい．どの様にしたらよいか？
- さらに，文 $\gamma = \forall X, Y, Z$ (win(X,Y)∧win(Y,Z)⇒win(Z,X)) が，恒真，恒偽，充足可能のいずれかを判定したい．どの様にしたらよいか？なお，領域や定数記号の対応に関しては，以下のとおりとする
  - 領域D = { グー, チョキ, パー }
  - 定数記号：r(ock), p(aper), s(cissors)の対応：r→グー，p→パー，s→チョキ
  - 関数記号：なし
  - 述語記号：win(X, Y) .. ジャンケンにおいて，手Xは手Yに勝つ

まとめ：今回の授業の内容

- 復習
  - 論理学における世界の表現
  - 命題論理
- 述語論理
  - 構文
  - 解釈
  - 分類

用語のまとめ

- 項：オブジェクトの論理表現
  - 関数記号（function symbol）：構造を持った複雑なオブジェクトを表現するための手段
  - 複合項（compound term）：関数記号を伴う項
  - 基礎項（ground term）：変数を全く含まない項
- 文
  - 原子文（atom, atomic sentence）：オブジェクト間の関係，オブジェクトの性質の論理表現
  - 基礎原子文（ground atom）：すべての引数が基礎項である原子文（変数を含まない原子文）
  - 複合文（complex sentence）：論理結合子を用いて原子文を連結した文
- 限量
  - 全称限量（universal quantifier）：すべてのオブジェクトに対する記述
  - 存在限量（existential quantifier）：あるオブジェクトに対する記述
  - 束縛変数（bound variable）：スコープ内にある変数・限量化されている変数
  - 閉論理式（閉じた文）：すべての変数が束縛されている述語文
- 解釈
  - 領域：述語文αが表現する世界での対象の集合
  - 解釈：述語文αの領域をDとする．このとき，αの解釈を以下のステップで与える
    - A中の定数記号，関数記号に，領域中の要素を対応させる
    - 対応関係の下で，基礎原子文への真理値の割り当てる
- 標準形
  - 冠頭標準形：式の左端（先頭）ですべての変数が限量された述語文
  - スコーレム無標準形：冠頭標準形＋「母式が連言標準形」＋「存在限量子を含まない」
  - 節集合：「すべての変数が左端で全称限量された節」の連言