

スーパーサイエンスハイスクール

人工知能プログラミング

AI programming

<http://www-wada.elcom.nitech.ac.jp/~inuzuka/SSHAiprogramming/>

講師 名古屋工業大学

犬塚信博

中野智文

助手 TA 近藤真一

今日の講義と演習のねらい

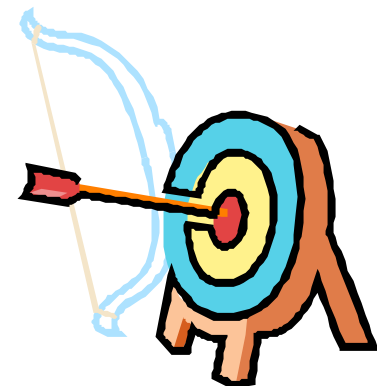
- 人工知能の基礎技術である論理と推論の働きについて理解する。
- 論理プログラムの基本を体験する。

キーワード

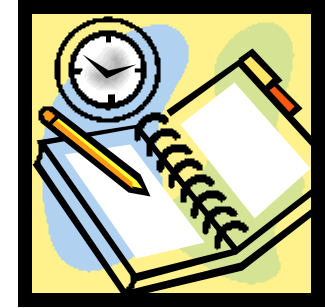
- 人工知能
- 論理、推論、知識
- 論理プログラム、Prolog

副作用

- 論理的思考を訓練する。



今日の予定



- 午前**
- 人工知能と論理の講義 演習
 - 論理、推論、正しい推論
 - Prologの入門
 - Prologをコンピュータに入れる、推論させる
- 午後**
- 人工知能の研究
 - データからの知識発見、帰納学習
 - Prologの演習
 - Prologを使って知識を書く、問題を解く

人工知能 (Artificial Intelligence)

- 機械的に人の持つ知能を実現する技術 (またはそうした技術の研究)



- 知能
 - ことばを話す、理解する、議論する。
 - 知識を使って考える (推論する、計画を立てる)
 - 学習する、科学する、技術をみかく。
 - 見てわかる。聞いてわかる。回りの状況を理解する。
-

人工知能の考え方

- 知識を中心に考える。



学習

知識を見つける、蓄える

知識

推論

知識を使う

言語

知識を交換する

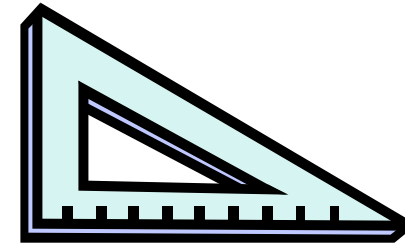
他にも考え方はあります。

- 脳細胞を中心に
(コネクショニズム)
- 身体と環境を中心に
(アフォーダンス)

命題 (propositions)

- 真偽のはっきりとわかる断定された文。

- 平行四辺形の向かい合う二辺の長さは等しい。
- 今朝、朝食をとった。
- 素数は無限個存在する。



- 命題でない文

- おはよう
- 今日は何日ですか？
- 毎日朝食をとりなさい。
- 神は存在する。
(もし、神が何であるかはっきり定義してないならば)

推論 (inference, reasoning)

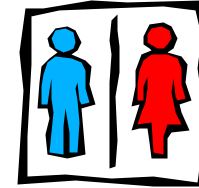
分かっていること(命題)から、新しい知識(命題)を導き出すこと。



- 演繹推論 : 確実に結論できる仕方で新しいことを導く。
- 帰納推論 : 個別の事実を総合して、一般法則を導く。
- 類推 : 分かっていることとの類似に注目して、新しい知識を導く。

いつも安心して使えるのは、演繹推論のみ。

推論 (演繹推論) の例 1



(1) 小泉首相は男性あるいは女性
のどちらかである。

(2) 小泉首相は女性ではない。

仮定
(前提)

(3) 小泉首相は男性である。

結論

推論 (演繹推論) の例 1

(1) P または Q

(2) Q でない

(3) P

P = 小泉首相は男性である

Q = 小泉首相は女性である

推論 (演繹推論) の例 2

(4) エンジンがかかるならば、燃料は入っている。

(5) エンジンがかからない。

(6) 燃料が入っていない。

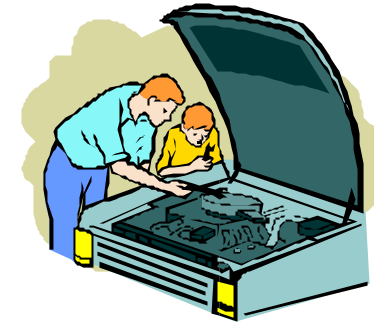


推論 (演繹推論) の例 2

(1) P ならば Q

(2) Q でない

(3) P でない



P = 燃料がはいっている

Q = エンジンがかかる

命題論理 (propositional logic)

- 命題を組合せて命題を作ることができる。
- 命題の組合せの正しさで、推論の正しさを考える論理を命題論理という。

命題の組合せ方

Pでない	PまたはQ	PかつQ	PならばQ
$\neg P$	P Q	P Q	P Q
否定	論理和(連言)	論理積(宣言)	含意

組合わせた命題の真偽

- 命題を組合わせてできた命題の真偽は、元の命題の真偽で決まる。
- $P \wedge Q$ (PかつQ): PとQが両方真であるとき、真。

P	Q	P	Q	P	Q	P	Q
真	真	真	真	真	真	真	真
真	偽	真	真	真	偽	真	偽
偽	真	真	真	偽	偽	真	真
偽	偽	真	偽	偽	偽	真	真

P	$\neg P$
真	偽
偽	真

この表を真理値表という



正しい推論

- 仮定が正しいならば、導かれた結論がいつも正しい形式の推論を正しい推論という。

A君は野球部員である。
A君は理系クラスである。

A君は野球部員で、理系
クラスである。

$$\frac{P \quad Q}{P \quad Q}$$


P	Q	P	Q
真	真	真	真
真	偽	真	偽
偽	真	偽	偽
偽	偽	偽	偽

- 推論の正しさは、形式でのみ決まる。
- 話の内容には関係ない。

正しい推論の例

小泉首相は男性あるいは女性
のどちらかである。

小泉首相は女性ではない。

小泉首相は男性である。

P	Q
$\neg Q$	
<hr/>	
P	

結論		仮定 1		仮定 2
↓		↓		↓
P	Q	P	Q	$\neg Q$
真	真	真		偽
真	偽	真		真
偽	真	真		偽
偽	偽	偽		真

確かに、仮定 1 と仮定 2 が正しい
ときには結論が常に正しい。

正しくない推論の例

エンジンがかかるならば、
燃料が入っている。

エンジンがかからない。

燃料が入っていない。

P	Q
$\neg P$	
<hr/>	
$\neg Q$	

		仮定 1	仮定 2	結論	
		↓	↓	↓	
P	Q	P	Q	$\neg P$	$\neg Q$
真	真	真		偽	偽
真	偽	偽		偽	真
偽	真	真		真	偽
偽	偽	真		真	真

仮定 1 と仮定 2 が真でも、結論が真でない場合がある。

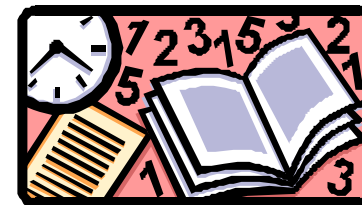
注意 推論の正しさは、実際に書く命題が正しいかどうかとは無関係。

練習 次の推論は正しいかどうか、
確かめなさい。

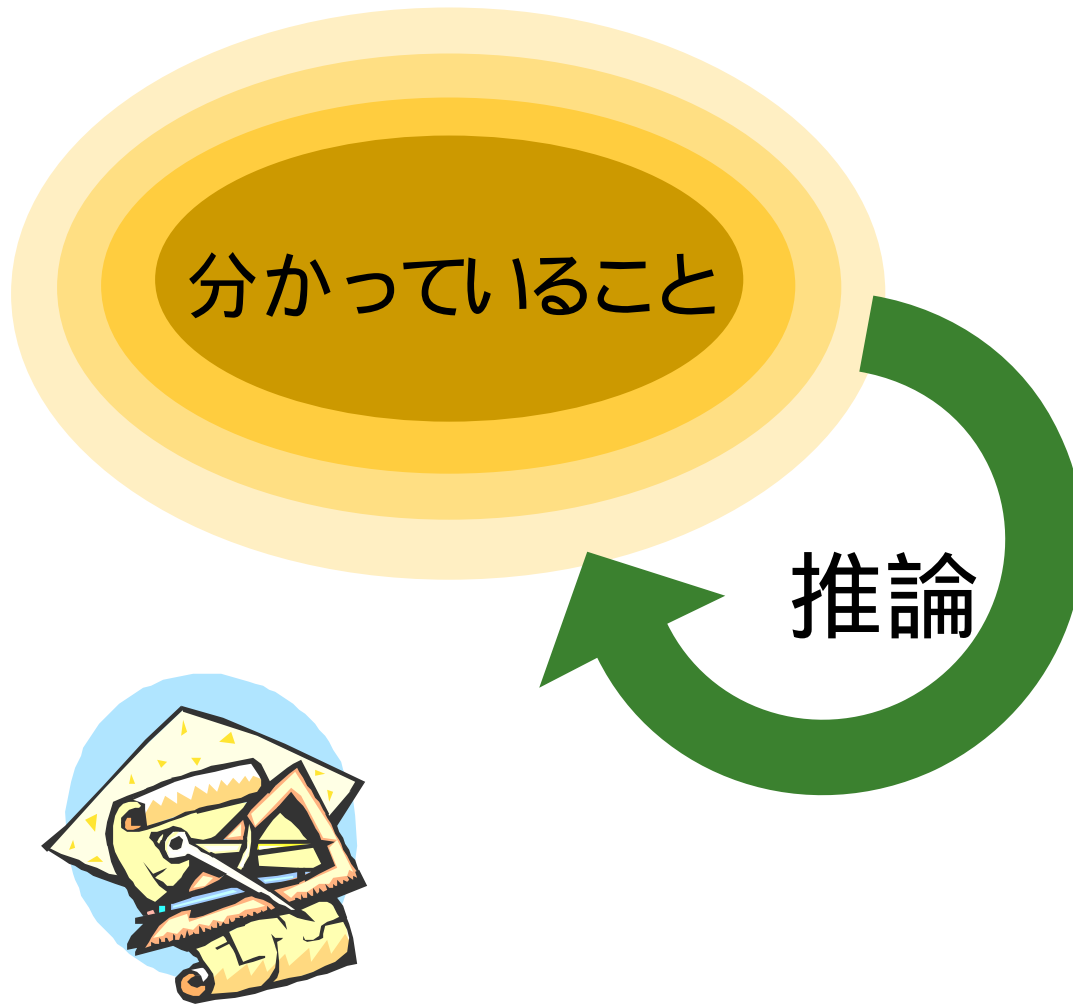
- (1) 雨が降る日はA先生は青いネクタイをしてくる。
A先生は青いネクタイをしていない。
だから、今日は雨は降らないだろう。



- (2) レポートを提出するか、試験が70点以上で
あれば合格である。
レポートを提出した。
だから、合格である。



公理と理論



- 推論で分かることを増やす。
- 最初にあった分かっていること
= 文句なしに認められること
= **公理**あるいは**理論**
axiom **theory**

例 ユークリッドの幾何の公理
ニュートンの力学の理論

論理と理論

論理と理論は見た目が似ていますが、全く違います。

- **論理** = 分かっていることから、新しいことを導くための**考え方**。
logic
- **理論** = ものごと(たとえば、経済、物理)を**説明**するため、
theory はじめに皆で認めることにした事実(仮説)

十分強力な論理があれば、理論からその帰結を知ることができる。

命題論理の限界

- 命題論理、命題の組合せでのみ結論を導き出しますが、それではつぎの推論は扱えません。

すべての人間は死ぬ。	P
ソクラテスは人間である。	Q
<hr/>	
ソクラテスは死ぬ。	R

どの命題も基本命題の組合せでできているわけではない。

命題論理の限界

だれでも人間は死ぬ。
ソクラテスは人間である。

ソクラテスは死ぬ。



これを扱うために、命題を主語と述語に分解して扱う

= 述語論理 (predicate logic)

ソクラテスは人間である
人間 (ソクラテス)

述語論理

- 述語論理では、変数を使います。

x は人間である = 人間 (x)

- 人間は死ぬ

= あるものは人間であるならば、そのものは死ぬ

= 「 x が人間ならば、 x は死ぬ」がどんな x にも成立つ。

これを、 $x. (\text{人間}(x) \rightarrow \text{死ぬ}(x))$ と書きます。

x は、 x に何を代入しても真であることを表します。

を全称記号とします。

述語論理

- 述語論理では、変数を使った論理のために2種類の記号を使います。
- **:全称記号** (allのAの逆)
 $x. P(x)$: どんなxに対してもP(x)は真である。
- **:存在記号** (existのEの逆)
 $x. P(x)$: P(x)が真であるようなxが存在する。

例 どんな人間も死ぬ $x. (\text{Human}(x) \rightarrow \text{Mortal}(x))$

 死なない人間がいる $x. (\text{Human}(x) \wedge \neg \text{Mortal}(x))$

練習 述語論理で表す

次の命題の意味が分かりますか？

ただし、Human(x): xは人間、

Like(x,y): xはyを好む、とします。

- $x. \{ \text{Human}(x) \quad y. [\text{Human}(y) \quad \text{Like}(x,y)] \}$
- $\neg \{ y. [\text{Human}(y) \quad (x. \text{Human}(x) \quad \text{Like}(x,y))] \}$

述語論理での推論

- 述語論理の推論の正しさは、真理値表では確かめられませんが、次の形式は正しい推論です。

すべての人間は死ぬ。
ソクラテスは人間である。

ソクラテスは死ぬ

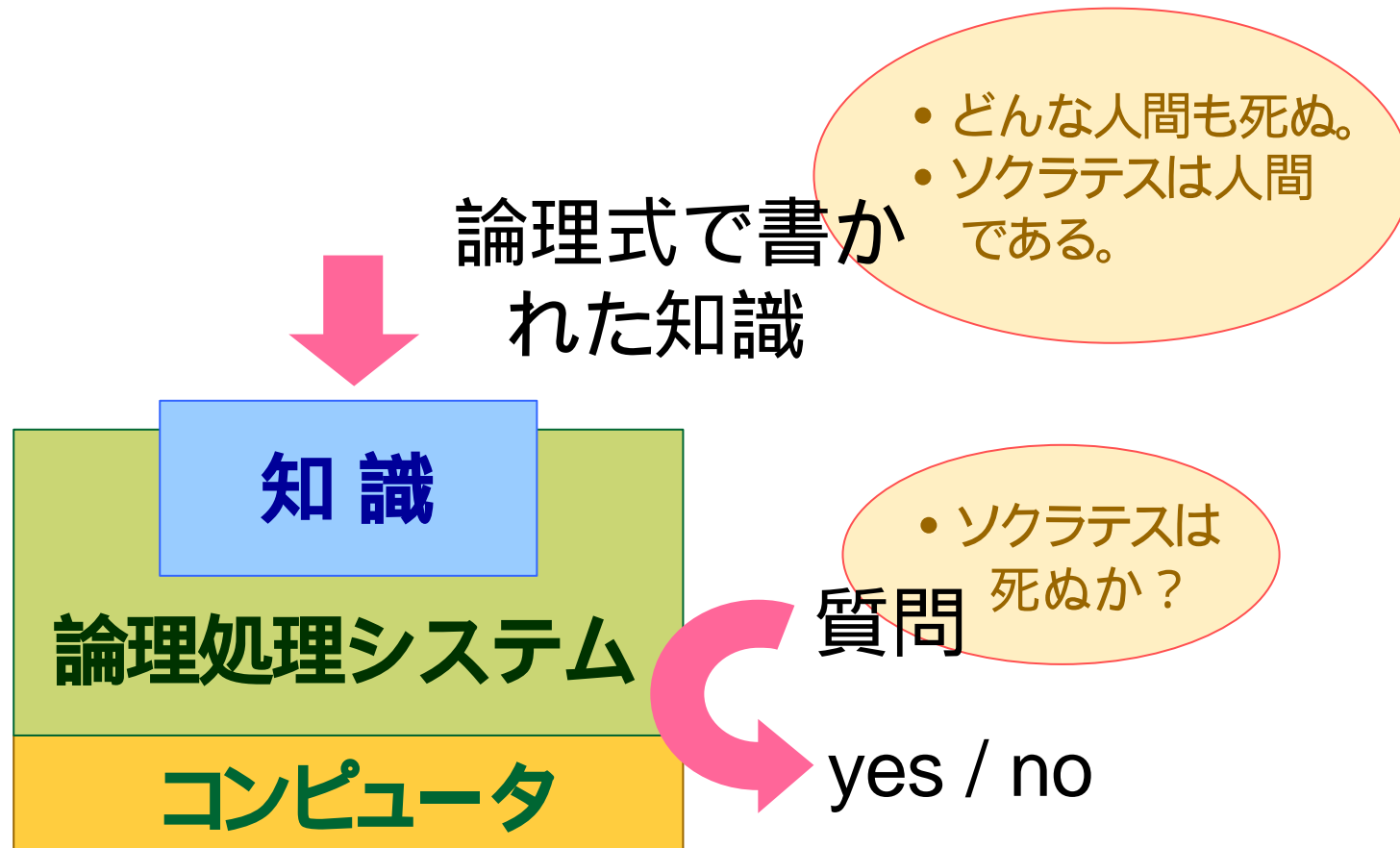
$$\begin{array}{l} \forall x. p(x) \rightarrow q(x) \\ p(S) \\ \hline q(S) \end{array}$$

すべての動物は死ぬ。
すべての人間は動物である。

すべての人間は死ぬ。

$$\begin{array}{l} \forall x. p(x) \rightarrow q(x) \\ \forall x. r(x) \rightarrow p(x) \\ \hline \forall x. r(x) \rightarrow q(x) \end{array}$$

論理プログラム (logic programs)



- 論理処理システムとして、Prologを用います。

論理プログラムの形式

Prolog : Programming in logic

述語論理の命題をコンピュータで扱うことができるようにしたシステム。

今日の演習ではSWI-Prologを使います。

SWI-Prolog :

オランダ アムステル大学が開発した無料のシステム。

Prologの記法 (1)

x. (Human(x) Mortal(x))

をprologで書くと

mortal(X) :- human(X).

- 「`:-`」は、逆の「`:-`」の意味で「`:-`」と書く。
- 全称記号は書かない。
(変数にはすべて全称記号がついていると思う)
- 述語は小文字で始める。
- 変数は大文字で書く。
- 最後はピリオド。

Prologの記法 (2)

一般にprologは次の形式の命題を許す。
これを論理プログラムでは「節 (せつ)」と呼びます。

$p(t_1, \dots, t_n)$.

$p(t_1, \dots, t_n) :- q_1(\dots), q_2(\dots), \dots, q_m(\dots)$.

- t_i は定数 (定項) (ex. socrates) か、変数 (ex. X)。
- 1つ目の形式は**事実**、2つ目は**規則 (ルール)**とします。
- 規則の右側 (条件) は、「かつ」で繋がっている。
- $\neg A$ は **not(A)** と書く。他の接続詞は基本的に使わない。

Prologの節

事実 ~である。

`human(socrates).`

ソクラテスは人間である。

`parent(taro,ichiro).`

タロウは一郎の親である。

ルール ~ ならば ~ である。

~ であるには ~ である必要がある。

`father(X,Y): -parent(X,Y), male(X).`

XがYの親であり、Xが男性ならば、XはYの父親である。

XはYの父親であるには、XがYの親で、Xが男性である必要がある。

Prologで推論する例

仮定の命題

```
mortal(X) :- human(X).  
human(socrates).
```

質問

```
?- mortal(socrates).
```

Prologで推論する例 (つづき)

- 質問をルールや事実とマッチングしてゆきます。

質問

?- mortal(**socrates**).

↕ マッチング

mortal(**X**) :- human(**X**).

- mortal(X)の条件はhuman (X)なので、これを次に質問します。
- マッチングでX=socratesになったので、新しい質問は、

新たな質問

?- human(**socrates**).

↕ マッチング

human(**socrates**).

Prologをコンピュータにインストールする

1. SWI-Prologのサイトをインターネットで検索する。
2. SWI-Prologのファイルをダウンロードする。
3. パソコンにインストールする。



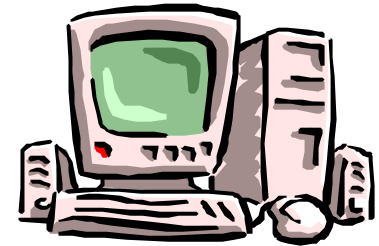
プログラミングの基本ステップ

1. 方法 (アルゴリズム、理論) の設計
2. コーディング (プログラムを書く)
3. エディット (編集) edit
4. 実行 run
5. デバッグ (誤り探し) debug
 1 ~ 3に戻る



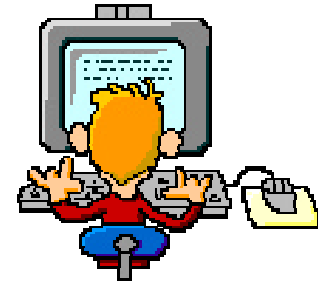
Prologを使ってみる(1)

1. 理論となる節 (命題) をファイルに書く。
2. ファイルを読み込む。
3. 質問の命題を入れて見る。
4. 上手くいかなければ、もう一度エディット



Prologを使ってみる (2)

1. 理論となる節 (命題) をファイルに書く。
 - i. File > New からファイルを開く。
適当な名前を付ける。 `file.pl`
 - ii. ファイルに節 (命題) を書く。
 - iii. セーブする。エディタの Save buffer



Prologを使ってみる (3)

2. ファイルを読み込む。

iv. File > Consult

読み込むと 節 (命題) は仮定された命題となります。

3. 質問の命題を入れて見る。

?- mortal(socrates).

仮定された知識から推論された答えを出します。

Prologを使ってみる(4)

4. 上手くいかなければ、もう一度エディット

v. File > Edit から、
で付けた名前のファイル
を呼び出す。

vi. から繰り返し。



練習 :Prologで推論してみる

質問に答えられるよう 事実をPrologの節として与えよ。
明示的に書かれていないことも必要なら加えよ。

事実

- タロウはヨウコにセーターを贈った。
- タカシはコーラを買った。

質問

- ヨウコは洋服を持っていますか？
- 飲み物を持っているのは誰ですか？



練習のヒント

事実

- タロウはヨウコにセーターを贈った。(gave)
- タカシはコーラを買った。(bought)

書いてないけど必要なこと

- AがBにCを贈ったら、BがCを持っている。(have)
- AがBを買ったら、AはBを持っている。
- セーターは洋服である。
- コーラは飲みものである。
- AがBであり(isa)、CがAを持っていれば、CはBを持っている。

Prologを使った数値計算

計算結果を出すには「X is 式」で書きます。

?- X is 2*4+6 / (1+2).

?- X is sin(pi / 6).

不等式も書けます。

?- 2+3 >= 1.



数値を計算する関数を論理的に書く

関数を定義してみます。

$$y = 2x^2 + 5$$

$$f(X, Y) :- Y \text{ is } 2 * X * X + 5$$

漸化式のような定義 (再帰的定義)

$$\begin{cases} 1! = 1 \\ n! = (n - 1)! * n \end{cases}$$

$$\text{fact}(1, 1).$$

$$\text{fact}(X, Y) :- X1 \text{ is } X - 1, \text{fact}(X1, Y1), Y \text{ is } Y1 * X.$$



練習課題 1 : 家族

家族に関する述語についての規則 (定義) を与えてください。

父、母、祖父、おば

`father(X, Y)`, `mother`, `grandFather`, `aunt`

兄弟、姉妹、いとこ、一人っ子

`brother(X, Y)`, `sister`, `cousin`, `anOnlyChild(X)`

祖先、子孫

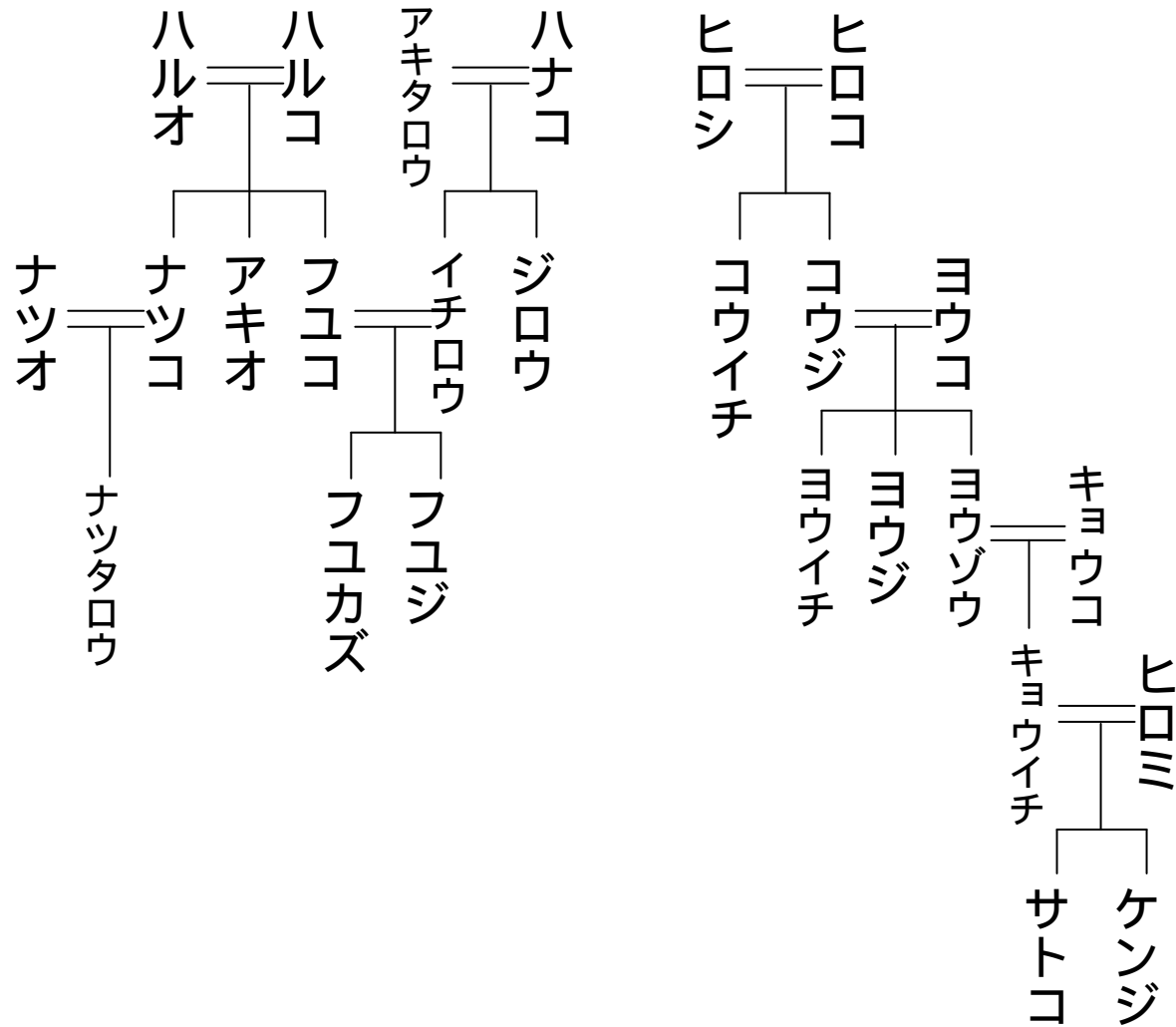
`ancestor(X, Y)`, `offspring(X, Y)`

家族 : 与えられている事実

- 親子関係
 - parent(taro, hanako).
タロウはハナコの親である。
- 性別
 - male(taro).
タロウは男である。
 - female(hanako).
ハナコは女である。

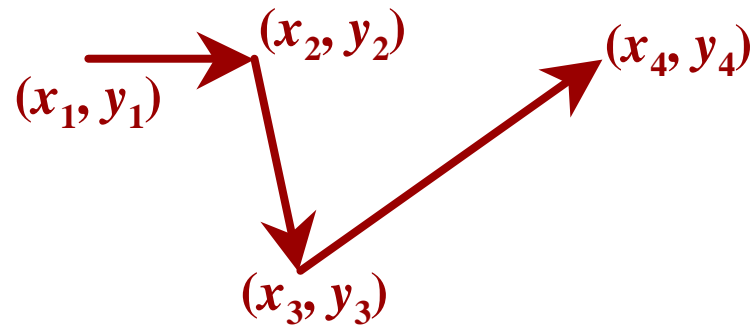
練習課題はこれらの事実で決まる関係です。

家族のデータ



練習課題 2 : 矢印の連結

点と点を矢印で結んだ図を考える。



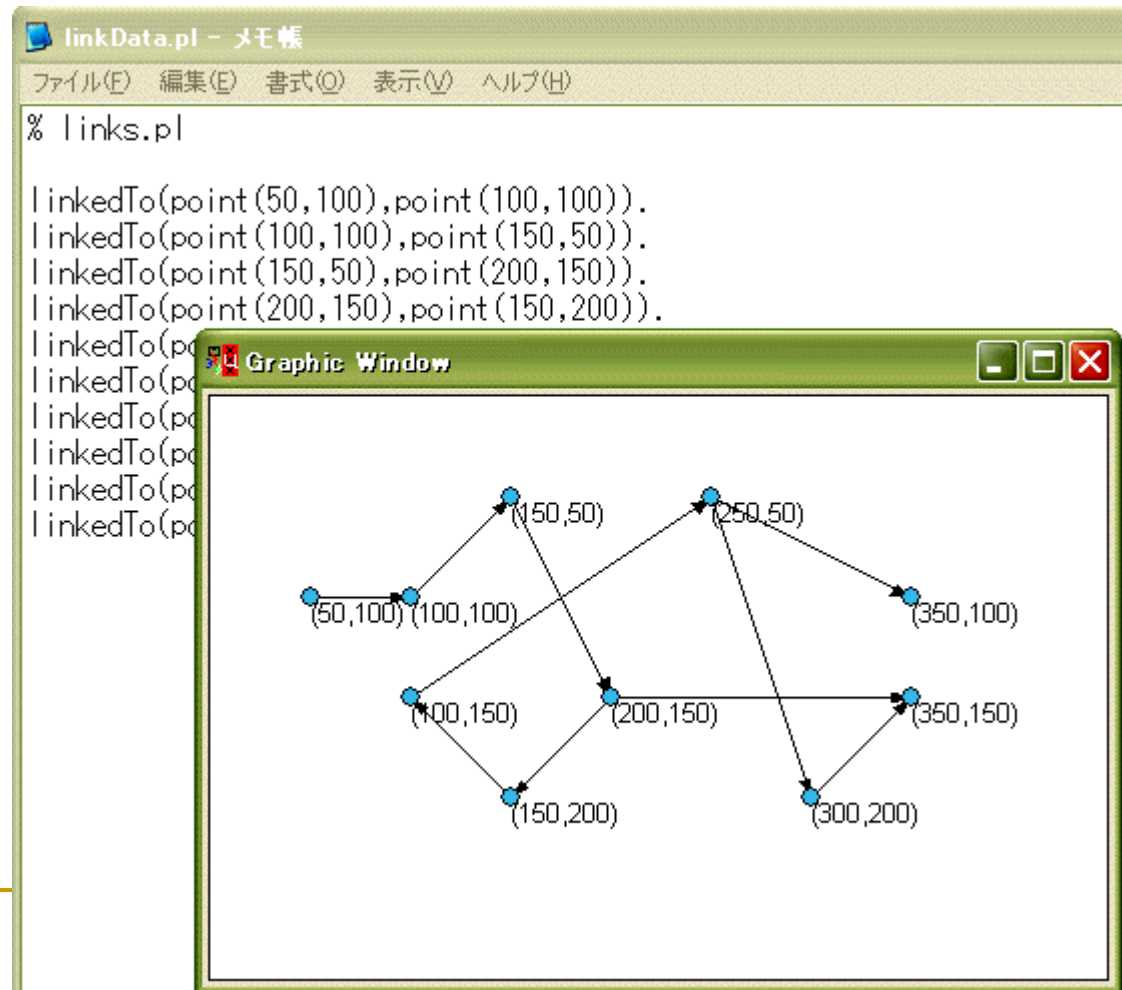
矢印で、2つの点 (x, y) と (x', y') が結ばれていることを次のように書くとする。

`linkedTo(point(x,y),point(x',y'))`.

描画のシステム

矢印を画面に出すプログラムを用意しています。

- graphic.plをconsultする。
- linkedToの関係のデータをconsultする。
- ?-drawArrows.で描画する。



課題

- 次の述語を定義してください。
点 (x, y) から矢印を通して点 (x', y') へ達することができるとき真となる述語
`canReach(point(x,y),point(x',y'))`.
点 (x, y) から矢印を通して右の点 (x', y') へ一度も左に戻らずに達することができるとき真となる述語
`doesNotBack(point(x,y),point(x',y'))`.

練習課題 3

次の述語で図形が座標上にあることを表します。

- 各辺が座標軸に平行な長方形 (矩形)
(左上の点と右下の点で表す)

`rectangle(shikaku1, point(x,y), point(x',y'))`.

- 円 (中心点と半径で表す)

`circle(en1, point(x,y), r)`.

課題

- 次の述語を定義してください。
2つの長方形、2つの円が重なりがあるときに真となる述語。

`intersect(shikaku1, shikaku2).`

`intersect(en1, en2).`

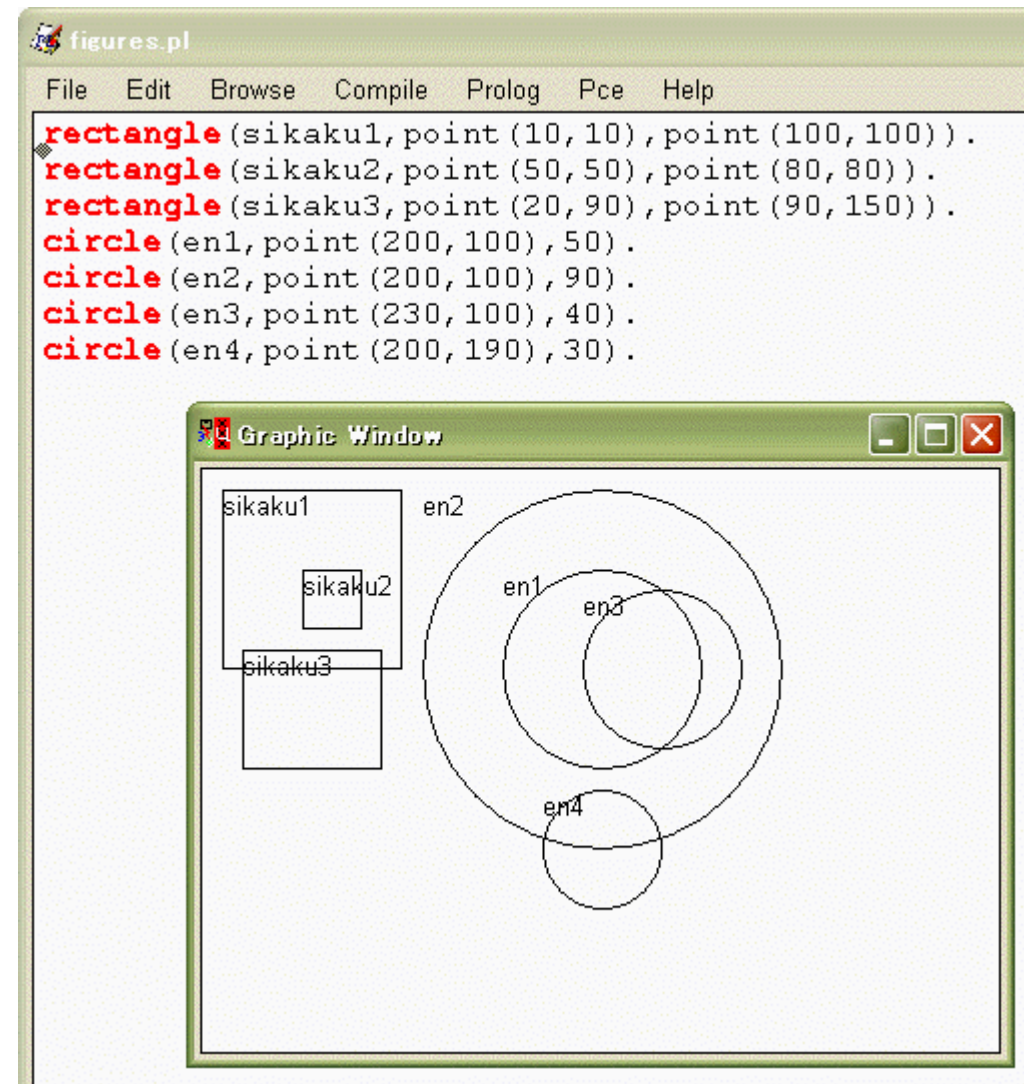
ある長方形 (円) が別の長方形 (円) に含まれるとき真となる述語。

`included(shikaku1, shikaku2).`

`included(en1, en2).`

図形の描画

- graphic.plで描画
できます。
- graphic.plと図形
データをconsult。
- ?-draw.で描画
できます。



```
figures.pl
File Edit Browse Compile Prolog Pce Help
rectangle(sikaku1, point(10, 10), point(100, 100)).
rectangle(sikaku2, point(50, 50), point(80, 80)).
rectangle(sikaku3, point(20, 90), point(90, 150)).
circle(en1, point(200, 100), 50).
circle(en2, point(200, 100), 90).
circle(en3, point(230, 100), 40).
circle(en4, point(200, 190), 30).
```

The screenshot shows a Prolog environment window titled 'figures.pl' with a menu bar (File, Edit, Browse, Compile, Prolog, Pce, Help) and a code editor containing the Prolog code above. Below the code editor is a 'Graphic Window' displaying the visual output of the code. The window shows three rectangles labeled 'sikaku1', 'sikaku2', and 'sikaku3' on the left, and four overlapping circles labeled 'en1', 'en2', 'en3', and 'en4' on the right. The rectangles are drawn with black outlines, and the circles are also drawn with black outlines. The circles 'en1' and 'en2' are concentric, with 'en1' being the inner one and 'en2' being the outer one. 'en3' is a smaller circle overlapping the right side of 'en1'. 'en4' is a small circle overlapping the bottom of 'en1'.

練習課題 4

- 文字を線分の集まりで表現します。
- 文字moji1を構成する線分が座標平面上にあることを次のように表す。

`line(moji1, point(x, y), point(x', y'))`.

たとえばLならば次のようになるでしょう。

`line(moji2, point(50, 50), point(50, 150))`.

`line(moji2, point(50, 150), point(150, 150))`.

課題

- いろいろな文字を認識する述語を作りたい。
- たとえば、?-letterA(moji1). で、moji1がAと読めるときに真となるようにしたい。
- 4種類の文字データを用意しました。

letterData1 : L, T, X。

letterData2 : A, E, W, Z。

letterData3 : A, B, C, D, E。

letterData4 : A ~ Zの26文字。

文字データの描画

- 文字もgraphic.plで描画できます。
- ?-draw.で描画。



The screenshot shows a Prolog environment with two windows. The top window, titled "Graphic Window", displays the letters "L T X" drawn on a white background. The bottom window, titled "letterdata1.pl", shows the following Prolog code:

```
File Edit Browse Compile Prolog Pce Help
line(l, point(100, 100), point(100, 140)).
line(l, point(100, 140), point(130, 140)).
line(t, point(150, 100), point(180, 100)).
line(t, point(165, 100), point(165, 140)).
line(x, point(200, 100), point(230, 140)).
line(x, point(230, 100), point(200, 140)).
▲
```

課題のヒント

次のような述語を用意するとよいかもしれません。

- 線分が鉛直である、水平である。
 - 線分が直角に接続している。
 - 線分がなめらかに (鈍角で)あるいは鋭く(鋭角で)接続している。
-