Introduction to Prolog

IF1221 Computational Logic
Semester II - 2024/2025

Informatics Engineering Study Program
School of Electrical Engineering and Informatics ITB

Introduction

- Nama PROLOG adalah singkatan dari *PROgramming in LOGic*.
- Bahasa ini pertama kali dikembangkan oleh Alain Colmerauer dan P. Roussel di Universitas Marseilles Perancis pada tahun 1972
- Dikembangkan berdasarkan Kalkulus Predikat Orde Pertama (Relational Logic)

Struktur Data

- Pemrograman dalam prolog meliputi tiga hal utama, yaitu:
  - Deklarasi sejumlah **fakta** mengenai suatu objek dan relasinya.
  - Pendefinisian sejumlah **rules/aturan** mengenai suatu objek dan relasinya.
  - **Pertanyaan** mengenai suatu objek dan relasinya.

- Struktur data PROLOG terdiri dari struktur data primitif berupa struktur data simbolik dan struktur data kompleks berupa list

Input

- Jenis masukan pada sistem:
  - Clause / Klausa
    - Program dari Prolog
    - Terdiri dari fakta dan aturan
    - Biasa disimpan dalam satu file
  - Directive
    - Aksi untuk dilakukan
  - Query
    - Request ke database
    - Usaha untuk mendeduksi apakah query tersebut bernilai true atau false

Clause / Klausa

- Fact / Fakta
  - kotak(spongebob).
  - kuning(spongebob).
  - teman(patrick,spongebob).

- Rules / Aturan
  - teman(X,Y) :- teman(Y,X)
  - spons(X):- kotak(X), kuning(X).

Fakta

- Fakta dituliskan dalam sebuah predikat untuk menggambarkan relationship
- Argumen dari fakta adalah konstanta yang terkait
- Misalnya : warna(langit, biru)
  - warna adalah sebuah predikat yang bebas didefinisikan oleh user
  - langit dan biru adalah konstanta yang terkait
  - Predikat dan konstanta ditulis dimulai dengan huruf kecil

Fakta

- Predikat dan parameter diawali dengan huruf kecil.
- Parameter predikat diawali oleh tanda kurung.
- Tanda titik '.' harus diberikan di akhir sebuah fakta.

Rules

- Rules/ aturan adalah akan memberi tahu sistem prolog bagaimana mendeduksi fakta baru yang tidak terdaftar di database secara eksplisit
- Aturan ditulis dalam bentuk implikasi.
- Bentuk aturan adalah P :- Q, R, S.

  yaitu P bernilai True jika Q, R dan S bernilai True. P adalah head sedangkan Q,R,S adalah body
- ‘,’ akan mengekspresikan dan. Untuk mengekspresikan atau, buat rule lain yang sama headnya namun berbeda body dari rule.

Rules

- Contoh :

  nenek(X,Y) :- ibu(X,Z),ibu(Z,Y).

  nenek(X,Y) :- ibu(X,Z),ayah(Z,Y).

GNU Prolog

- ***GNU Prolog interactive interpreter***
- *GNU Prolog native-code compiler*

Directive

- Misal untuk meload program dari sebuah file

  | ?- [facts].

  | ?- consult('facts.pl').

- Menerima masukan klausa interaktif dari user

  | ?- [user].

- Untuk keluar dari prolog

  | ?- halt.

Query

- Menanyakan sejumlah pertanyaan mengenai fakta yang telah didefinisikan.
- Meminta prolog untuk menentukan sebuah query bernilai true atau false
  - | ?- ayah(berta, philips).
  - | ?- ayah(berta, X).

Identifier

- Predikat/Fungsi (diawali dengan huruf kecil)
  - Hanya boleh mengandung karakter [A-Z, a-z, 0-9, _]
  - Case sensitive
  - Contoh: ayah, pria, kakek, kuning
- Konstanta/Objek (diawali dengan huruf kecil)
  - Hanya boleh mengandung karakter [a-z, 0-9, _]
  - Case sensitive
  - Contoh: philips, dicky, rumah, spongebob
- Atom adalah berupa konstanta atau string.

  philips,"albert" adalah sebuah atom

Variable Anonymous

- Variable (diawali dengan huruf besar, atau diawali dengan _)
  - X, Y, Z, Problem, _variable, _var
- Variable _ untuk variable anonymous
  - makan(Godzilla,_).
  - Dapat diartikan "godzilla memakan semuanya". '_' bisa digantikan dengan objek apapun
- '_' juga bisa digunakan untuk variable yang tidak perlu diakses
  - Hanya boleh mengandung karakter [A-Z, a-z, 0-9, _]
  - Case sensitive

Tipe Lain

- Komentar : /* adalah sebuah komentar */
- Numeric : 1 , 2.3
- Karakter : ‘a’ , ’B’
- String : “adalah sebuah string”

Operator

Ekspresi	Arti
X+Y	Penambahan
X-Y	Pengurangan
X*Y	Perkalian
X/Y	Pembagian
X//Y	Pembagian Integer
X mod Y	Modulo
-X	Negasi
[X]	Dievaluasi ke X jika X adalah integer

Ekspresi	Arti
X = Y	Mengaitkan X dangan Y
Z is X	Mengaitkan Z dengan hasil evaluasi X (Aritmetik)
X==Y	Menyatakan nilai X dan Y adalah sama
X=:=Y	Menyatakan nilai X dan Y adalah sama (aritmetik)
X\==Y	Manyatakan nilai X dan Y adalah tidak sama
X=\=Y	Menyatakan nilai X dan Y adalah tidak sama (aritmetik)
X>Y	Menyatakan nilai X lebih besar dari Y
X<Y	Menyatakan nilai X kurang dari Y
X>=Y	Menyatakan nilai X lebih dari sama dengan Y
X=<Y	Menyatakan nilai X kurang dari sama dengan Y

Aritmetika

- Contoh:

  |?- X is 1+2.

  - X = 3.

  |?- X is 1+2, X<4.

  - X=3
  - yes.

Struktur Program

- Organisasi cara pertama, fakta dan aturan dikelompokkan secara **terpisah**. Seluruh fakta dikelompokkan tersendiri, demikian juga aturan dikelompokkan tersendiri, tetapi penulisannya fakta mendahului aturan.

- Organisasi cara kedua, fakta dan aturan dikelompokkan secara **tidak terpisah**. Pengelompokan didasarkan hanya atas nama predikat yang sama, atau lebih luasnya: didasarkan atas sekelompok fakta dan aturan yang mewakili suatu persoalan. Tetapi dalam penulisannya tetap fakta mendahului aturan.

Struktur Program

Cara Pertama	Cara Kedua
/* FAKTA */ /* Spesifikasi Fakta ke-1 */ ..... /* Spesifikasi Fakta ke-n */ ..... /* ATURAN */ /* Spesifikasi Aturan ke-1 */ ..... /* Spesifikasi Aturan ke-n */	/* FAKTA */ /* Spesifikasi Fakta ke-1 */ ..... /* Spesifikasi Fakta ke-n */ ..... /* ATURAN */ /* Spesifikasi Aturan ke-1 */ ..... /* Spesifikasi Aturan ke-n */ ..... /* FAKTA DAN ATURAN */ /*Spesifikasi Fakta & Aturan ke-1 */ ..... /*Spesifikasi Fakta & Aturan ke-n */ .....

Contoh Pengorganisasian yang Baik

Cara Pertama	Cara Kedua
PROGRAM POHON KELUARGA INGGRIS	**PROGRAM ARITMATIKA**

```
/* FAKTA */
/* pria(X) benar, jika X adalah
pria */
  pria(philip).
  pria(charles).
  pria(williams).
/* ayah(X,Y) benar, jika X adalah
ayah dari Y */
  ayah(philip,charles).
  ayah(charles,williams).
  ayah(charles,harry).
/* ATURAN */
/* anakLelaki(X,Y) benar, jika Y
adalah ayah dari X dan X adalah
pria dan X tidak sama dengan Y*/
  anakLelaki(X,Y) :- ayah(Y,X),
                     pria(X), X <> Y.
```

```
/* FAKTA */
/* suksesor(X,Y) benar, jika Y
adalah suksesor dari X */
  suksesor(0,1).
  suksesor(1,2).
  suksesor(2,3).
  suksesor(3,4).
/* ATURAN */
/* max2(X,Y,Z) benar, jika Z
adalah nilai maksimum dari 2
bilangan X dan Y */
   max2(X,Y,X) :- X >= Y.
   max2(X,Y,Y) :- X < Y.
/* FAKTA DAN ATURAN */
/* plus(X,Y,Z) benar, jika Z
adalah hasil penjumlahan dari X
dan Y */
/* Basis */
  plus(0,X,X).
/* Rekurens */
   plus(X,Y,Z) :- suksesor(V,X),
                  plus(V,Y,W),
                  suksesor(W,Z).
```

Eksekusi

- Prolog akan mencari fakta yang sesuai dengan melakukan unifikasi variable-variable
- Misal : query ayah(charles,W).
- ayah(charles,W) bertemu dengan fakta ayah(charles,williams) sehingga W akan diunifikasi dengan williams.

Gnu prolog

- Load fakta dan aturan dari file eksternal (*.pl), kemudian berikan query.

  | ?- consult (‘nama_direktori_tempat_file/namafile.pl’).

Pemrograman Deklaratif Murni

- Terdiri dari fakta dan aturan dalam bentuk kalkulus predikat orde satu.
- Predikat adalah benar-benar menunjukkan relasi antar objek yang menjadi parameter. Semua parameter adalah **parameter input**.
- Pada *body* dari aturan (ruas kanan tanda ← atau :-), karakter koma berarti operator AND secara logika, sehingga urut-urutan evaluasi tidak mempengaruhi hasil *query*.
- Komputasi program adalah murni pencocokan.
- Program hanya melakukan manipulasi/pencocokan terhadap **fakta simbolik**.
- *Cut* dan *fail* tidak ada gunanya (akan dibahas nanti).

Pemrograman Deklaratif Tidak Murni

- Terdapat kalkulasi dalam program deklaratif.
- Transformasi bentuk yang tidak mungkin dinyatakan dalam kalkulus predikat orde satu (seperti pada Plus) akan menjadi ***sequence***. Dalam hal ini tanda koma pada body aturan (di ruas kanan tanda ← atau :-) berarti urutan pencocokan aturan dan bukan AND secara logika. Jika urutan penulisan dibalik, hasil program akan "salah".
- Adanya parameter yang interpretasinya adalah **parameter output** (walaupun merupakan parameter dari predikat).
- Terdapat aspek prosedural dalam program
- *Cut* dan *fail* sangat penting karena mempengaruhi hasil *query* (akan dibahas nanti)

Analisis Kasus

- Penulisan kasus yang juga berupa predikat haruslah dalam bentuk fakta atau aturan
- Untuk menghindari ketergantungan eksekusi, setiap kasus haruslah dituliskan secara eksplisit, baik kondisi maupun aksi akibat kondisi tersebut
- Dalam pemrograman deklaratif sendiri, sebuah aturan secara tidak langsung adalah penulisan sebuah kasus
- Aturan tersebut mengandung kondisi yang harus dipenuhi agar aturan tersebut dibangkitkan (dipilih sebagai predikat untuk melakukan proses pencocokan saat dilakukan *query* pada program deklaratif).

Analisis Kasus

- Domain dalam analisis kasus: disjoint dan complete
- Kondisi dituliskan sebagai salah satu komponen dalam aturan
- Penulisan spesifikasi dari program deklaratif perlu dilakukan agar terhindar dari salah interpretasi
- Contoh:

```
/*max2(X,Y,X) benar, jika X >= Y
  max2(X,Y,Y) benar, jika X < Y */

  max2(X,Y,X) :- X >= Y.
  max2(X,Y,Y) :- X < Y.
```

Analisis Kasus

- Analisa kasus dapat merupakan ekspresi "ATAU" seperti pada contoh berikut

```
/* anakLelaki(X,Y) benar, jika  X adalah anak
  lelaki dari Y :
  Y adalah ayah dari X dan  X adalah pria dan Y
  adalah pria
  atau
  Y  adalah Ibu dari X dan X adalah pria dan Y
  adalah perempuan */
  anakLelaki(X,Y) :- ayah(Y,X), pria(X),
  pria(Y).
  anakLelaki(X,Y) :- ibu(Y,X), pria(X),
  perempuan(Y).
```

Review Prolog

- Program PROLOG terdiri dari fakta dan aturan. Sedangkan *query* dapat berada di dalam atau di luar program.
- Struktur data PROLOG terdiri dari struktur data primitif berupa struktur data simbolik dan struktur data kompleks berupa *list*
- Program deklaratif mengandung 'kasus' sebagai pemicu suatu aturan
- Kasus harus dituliskan dalam bentuk fakta atau aturan
- Kasus harus dituliskan secara eksplisit untuk menghindari ketergantungan eksekusi

Exercise Prolog

Jika terdapat fakta dan aturan sebagai berikut dalam Prolog:

```
ibu(emy, charles).
ibu(emy, david).
ibu(emy, randy).
ibu(maria, fara).
saudara_kandung(Anak1, Anak2) :- ibu(Ibu,
  Anak1), ibu(Ibu, Anak2).
```

a) Jika diberikan query saudara_kandung(Siapa1, Siapa2) dan kita menginginkan seluruh jawaban ditampilkan, apa jawaban prolog terhadap Siapa1 dan Siapa2.

b) Definisi di atas adalah masih terlalu sederhana, dan bila digunakan dalam sistem prolog, hasil query di atas tidak merepresentasikan masalah di dunia nyata. Ajukan perbaikan aturan di atas dan jelaskan mengapa perlu diajukan perbaikan tersebut

THANK YOU