STRUKTUR KOMPOSISI
(TIPE BENTUKAN)
DALAM BAHASA PROLOG

Bahasan

Pada bagian ini akan dibahas hal-hal berikut :

1. Pengertian Struktur Komposisi dalam Bahasa PROLOG
2. Penggunaan Functor
3. Contoh Kasus

Pengertian Struktur Komposisi dalam Bahasa PROLOG

Dalam Kalkulus Predikat, setiap predikat mempunyai sebuah nama dan satu/lebih argumen berupa term. Jenis term tersebut adalah : konstanta, variabel, dan fungsi. Fungsi sendiri dapat mempunyai nama dan satu/lebih argumen yang juga berupa term. Tetapi, jika nilai predikat adalah TRUE atau FALSE (benar atau salah; yes atau no) maka nilai fungsi adalah sebuah nilai hasil pemetaan fungsi (range) sesuai domainnya. Contohnya :

- konstanta : Ali, Manusia, dsb.
- variabel : Nilai, Hasil, X, Y, Ls, dsb.
- fungsi : f(X,Y), Ibu(Ani), dsb.
- predikat : Ibu(Ibu(Ani),Ali), LebihBesar(Z,f(X,Y)), LebihKecil(5,Kali(2,3)), dsb.

Bahasa PROLOG merupakan subset dari Kalkulus Predikat. Karena itu bentuknya lebih sederhana. Dalam bahasa PROLOG , setiap predikat juga mempunyai nama dan satu atau lebih argumen berupa term. Tetapi jenis termnya adalah sebagai berikut :

- konstanta : ditulis sebagai string yang diawali huruf kecil, misalnya : ali, manusia, dsb.
- variabel : ditulis sebagai string yang diawali huruf besar, misalnya : Nilai, Hasil, X, Y, Ls, dsb.
- fungsi : nama fungsi berupa string yang diawali huruf kecil, dengan satu atau lebih argumen berupa term, tetapi dengan pengertian lebih terbatas.

Pengertian fungsi dalam PROLOG :

- fungsi dikenal dengan istilah functor.
- fungsi tidak dapat mengembalikan nilai hasil pemetaan fungsi (range) sesuai domainnya.
- fungsi hanya dapat membentuk struktur komposisi dari term. Contohnya : buku(if-960013,'Clocksin and Mellish','Programming in PROLOG', 1984), jadwal(if224,dosen(IL,SP),waktu(Selasa,11:00,13:00),R9108), dsb.

Kecuali untuk proses aritmatika dalam PROLOG, yang mempunyai fungsi yang mengembalikan nilai hasil pemetaan fungsi (range) sesuai domainnya. Yaitu, fungsi-fungsi berikut yang notasi penulisannya sedikit berbeda karena berbentuk infix dan bukannya prefix :

NO	BENTUK UMUM	PROLOG
1	plus(X,Y)	X + Y
2	minus(X,Y)	X - Y
3	kali(X,Y)	X * Y
4	bagi(X,Y)	X / Y
5	div(X,Y)	X div Y
6	mod(X,Y)	X mod Y

Penggunaan Functor

Sebuah struktur komposisi dari term membutuhkan : konstruktor, selektor, dan predikat lain untuk memanipulasinya. Jika struktur komposisi tersebut ingin bernama, maka dapat digunakan functor. Tetapi, jika tidak diperlukan nama, maka struktur komposisi itu dapat berupa list linier dengan jumlah elemen sama dengan jumlah elemen struktur.

Bahasa PROLOG (standar : Clocksin&Mellish) menyediakan beberapa predikat sistem yang khusus digunakan untuk memanipulasi struktur komposisi yang menggunakan functor, yaitu :

1. functor(T,F,N) benar, jika T adalah struktur dengan functor F dan arity (jumlah argumen) N.
2. arg(N,T,A) benar, jika A adalah argumen ke-N dari struktur T.
3. univ(X,L), atau lebih sering ditulis sebagai **X=..L** , benar jika L adalah list yang berisi functor X diikuti dengan dengan seluruh argumen dari X.

Ketiga predikat itu dapat berlaku sebagai konstruktor dan selektor dari sebarang struktur.

Contoh pemakaian ketiga predikat :

1. *{copy(Old,New) benar, jika New adalah struktur baru yang merupakan copy dari struktur Old.}*

```
copy(Old,New) :- functor(Old,F,N), functor(New,F,N).
```

2.
```
arg(2,jadwal(if224,dosen(IL),waktu(Selasa,11,13),R9108),X)?
yes
X = dosen(IL)
```

3.
```
kuliah(if224,pengajar(IL,SP)) =.. X?
X = [kuliah,if224,pengajar(IL,SP)]
```

Dalam Turbo Prolog (hingga versi 2.0) predikat sistem tersebut tidak tersedia. Tetapi, Turbo Prolog dapat menangani pembentukan struktur komposisi tersebut dengan cara pendefinisian struktur dalam DOMAINS (sekaligus konstruksi). Lihat contoh kasus.

Contoh Kasus 1 : tipe dengan nama (versi-1)

PECAHAN : pecahan(n:integer,d:integer) versi-1

REALISASI DALAM TURBO PROLOG

```
/*-------------------------------------------------*/
/* STRUKTUR KOMPOSISI                              */
/* versi-1 : realisasi tipe dengan nama (implisit) */
/*-------------------------------------------------*/
/* Tipe pecahan : pecahan(n:integer,d:integer)     */
/*-------------------------------------------------*/
DOMAINS
  dpecahan = pecahan(dpemb,dpeny)
  dpemb,dpeny = integer

PREDICATES
  isPecahan(dpecahan)
  addP(dpecahan,dpecahan,dpecahan)
  subP(dpecahan,dpecahan,dpecahan)
  mulP(dpecahan,dpecahan,dpecahan)
  divP(dpecahan,dpecahan,dpecahan)
  realP(dpecahan,real)
  isEqP(dpecahan,dpecahan)
  isLtP(dpecahan,dpecahan)
  isGtP(dpecahan,dpecahan)

CLAUSES /* implisit */
  isPecahan(pecahan(_,D)) :- D > 0.

  addP(pecahan(N1,D1),pecahan(N2,D2),pecahan(N3,D3)) :-
     N3 = N1*D2 + N2*D1,
     D3 = D1 * D2.

  subP(pecahan(N1,D1),pecahan(N2,D2),pecahan(N3,D3)) :-
     N3 = N1*D2 - N2*D1,
     D3 = D1 * D2.

  mulP(pecahan(N1,D1),pecahan(N2,D2),pecahan(N3,D3)) :-
     N3 = N1 * N2,
     D3 = D1 * D2.

  divP(pecahan(N1,D1),pecahan(N2,D2),pecahan(N3,D3)) :-
     N3 = N1 * D2,
     D3 = D1 * N2.

  realP(P,R) :- R = N / D.

  isEqP(pecahan(N1,D1),pecahan(N2,D2)) :- N1 * D2 = D1 * N2.

  isLtP(pecahan(N1,D1),pecahan(N2,D2)) :- N1 * D2 < D1 * N2.

  isGtP(pecahan(N1,D1),pecahan(N2,D2)) :- N1 * D2 > D1 * N2.
```

Contoh Kasus 2 : tipe dengan nama (versi-2)

PECAHAN : pecahan(n:integer,d:integer) versi-2

REALISASI DALAM TURBO PROLOG

```
/*-------------------------------------------------*/
/* STRUKTUR KOMPOSISI                              */
/* versi-2 : realisasi tipe dengan nama (eksplisit) */
/*-------------------------------------------------*/
/* Konstruktor, selektor, dan predikat dari        */
/*   tipe pecahan : pecahan(n:integer,d:integer)   */
/*-------------------------------------------------*/
DOMAINS
  dpecahan = pecahan(dpemb,dpeny)
  dpemb,dpeny = integer

PREDICATES
  makePecahan(dpemb,dpeny,dpecahan)
  isPecahan(dpecahan)
  pemb(dpemb,dpecahan)
  peny(dpeny,dpecahan)
  addP(dpecahan,dpecahan,dpecahan)
  subP(dpecahan,dpecahan,dpecahan)
  mulP(dpecahan,dpecahan,dpecahan)
  divP(dpecahan,dpecahan,dpecahan)
  realP(dpecahan,real)
  isEqP(dpecahan,dpecahan)
  isLtP(dpecahan,dpecahan)
  isGtP(dpecahan,dpecahan)


CLAUSES  /* eksplisit */
  makePecahan(N,D,P) :- P = pecahan(N,D).

  isPecahan(P) :- P = pecahan(_,D), D > 0.

  pemb(N,P) :- P = pecahan(N,_).

  peny(D,P) :- P = pecahan(_,D).

  addP(P1,P2,P3) :- pemb(N1,P1), peny(D1,P1),
                    pemb(N2,P2), peny(D2,P2),
                    N3 = N1*D2 + N2*D1,
                    D3 = D1 * D2,
                    makePecahan(N3,D3,P3).

  subP(P1,P2,P3) :- pemb(N1,P1), peny(D1,P1),
                    pemb(N2,P2), peny(D2,P2),
                    N3 = N1*D2 - N2*D1,
                    D3 = D1 * D2,
                    makePecahan(N3,D3,P3).
```

```
mulP(P1,P2,P3) :- pemb(N1,P1), peny(D1,P1),
                  pemb(N2,P2), peny(D2,P2),
                  N3 = N1 * N2,
                  D3 = D1 * D2,
                  makePecahan(N3,D3,P3).

divP(P1,P2,P3) :- pemb(N1,P1), peny(D1,P1),
                  pemb(N2,P2), peny(D2,P2),
                  N3 = N1 * D2,
                  D3 = D1 * N2,
                  makePecahan(N3,D3,P3).

realP(P,R) :- pemb(N,P), peny(D,P),
              R = N / D.

isEqP(P1,P2) :- pemb(N1,P1), peny(D1,P1),
                pemb(N2,P2), peny(D2,P2),
                N1 * D2 = D1 * N2.

isLtP(P1,P2) :- pemb(N1,P1), peny(D1,P1),
                pemb(N2,P2), peny(D2,P2),
                N1 * D2 < D1 * N2.

isGtP(P1,P2) :- pemb(N1,P1), peny(D1,P1),
                pemb(N2,P2), peny(D2,P2),
                N1 * D2 > D1 * N2.
```

Contoh Kasus 3 : tipe tanpa nama; implementasi sebagai list

PECAHAN : [n:integer,d:integer]

REALISASI DALAM TURBO PROLOG

```
/*-------------------------------------------------*/
/* STRUKTUR KOMPOSISI                              */
/* versi-3 : realisasi tipe tanpa nama             */
/*-------------------------------------------------*/
/* Tipe pecahan : [n:integer,d:integer]            */
/*-------------------------------------------------*/
DOMAINS
  dpecahan = dlist
  dlist = delemen*
  delemen = integer

PREDICATES
  isPecahan(dpecahan)
  addP(dpecahan,dpecahan,dpecahan)
  subP(dpecahan,dpecahan,dpecahan)
  mulP(dpecahan,dpecahan,dpecahan)
  divP(dpecahan,dpecahan,dpecahan)
  realP(dpecahan,real)
  isEqP(dpecahan,dpecahan)
  isLtP(dpecahan,dpecahan)
  isGtP(dpecahan,dpecahan)

CLAUSES
  isPecahan([_|D]) :- D > 0.

  addP(pecahan(N1,D1),[N2|D2],[N3|D3]) :- N3 = N1*D2 + N2*D1,
                                          D3 = D1 * D2.

  subP([N1|D1],[N2|D2],[N3|D3]) :- N3 = N1*D2 - N2*D1,
                                   D3 = D1 * D2.

  mulP([N1|D1],[N2|D2],[N3|D3]) :- N3 = N1 * N2,
                                   D3 = D1 * D2.

  divP([N1|D1],[N2|D2],[N3|D3]) :- N3 = N1 * D2,
                                   D3 = D1 * N2.

  realP([N|D],R) :- R = N / D.

  isEqP([N1|D1],[N2|D2]) :- N1 * D2 = D1 * N2.

  isLtP([N1|D1],[N2|D2]) :- N1 * D2 < D1 * N2.

  isGtP([N1|D1],[N2|D2]) :- N1 * D2 > D1 * N2.
```