

1906002132015

Programlama Dilleri Temelleri

BAİBÜ Bilgisayar Müh.

Ders 10

Dr. Ö. Ü. İsmail Hakkı Parlak

Kaynaklar: Watt, David A., Programming Language Design Concepts, Wiley

Bazı şekiller kaynak kitaptan kopyalanmıştır.

PROLOG - Değişkenler

- Bir Prolog değişkeni, herhangi bir türden sabit ancak bilinmeyen bir değeri belirtir. Bu nedenle Prolog değişkenleri, zorunlu (imperative) bir dilin güncellenebilir değişkenlerine değil, matematiksel değişkenlere karşılık gelir.
- Bir değişken bir önermedeki görünümüyle deklare edilir ve kapsamı (scope) yalnızca içinde geçtiği önermedir.

```
icerde(nokta(X, Y), R) :- X*X+Y*Y < R*R.
```

```
?- icerde(nokta(3, 4), 6). % sorgusu başarılıdır.
```

```
?- X is 3 + 7.
```

```
X = 10
```

PROLOG - Aritmetik Operatörler

- Toplama (+)
- Çıkarma (-)
- Çarpma (*)
- Bölme (/)
- Üssü (**)
- Tam sayı bölmesi (//)
- Kalan (mod)
- Kare kök (sqrt)
- Maksimum (max)

```
?- K is 3, B is 5,  
   T is max(K, B).
```

PROLOG - Karşılaştırma Op.

- Büyüktür ($>$)
- Küçüktür ($<$)
- Büyük eşittir ($>=$)
- Küçük eşittir ($<=$)
- Eşittir ($=:=$)
- Eşit değildir ($=\backslash=$)

?- 1 + 4 =:= 2 + 3.

?- 1 + 4 =\= 7 * 3.

PROLOG - Birleştirim

- Prolog'un iki terimi eşleştirme şekline birleştirim (unification) denir. Örneğin:

baba(ali, veli).

?- baba(X, Y).

- şeklinde bir sorgu yaptığımızda X, ali'yi; Y, veli'yi temsil eder (instantiation). Burada baba(X, Y) terimi baba(ali, veli) ile **birleşmiştir** deriz. X, ali'ye; Y, veli'ye bağlanmıştır (binding).

PROLOG - Birleştirim

- Prolog'da birleştirimi gerçekleştirecek 2 terim (T_1 , T_2) verildiğinde:
- T_1 ve T_2 atom veya sayıysa (sabitler), T_1 , T_2 'ye eşitse birleştirim başarılı olur.
- T_1 bir değişkense, T_1 , T_2 'ye eşlenir.
- T_2 bir değişkense, T_2 , T_1 'e eşlenir.
- T_1 ve T_2 aynı parametre sayısına (**arity**) sahip kompleks terimlerse, örn:
 $T_1 = h_1(x_1, x_2, \dots, x_n)$, $T_2 = h_2(y_1, y_2, \dots, y_n)$, $h_1 = h_2$ ve $x_1 = y_1$, $x_2 = y_2$,
 \dots , $x_n = y_n$ ise birleştirim başarılı olur.
- Bunların hiç biri gerçekleşmediyse birleşim başarısız olur.

PROLOG - Listeler

- PROLOG'daki bir liste, bir yapının (structure) özel bir halidir. [] atomu boş listeyi, ".(x, xs)" yapısı ise başı x olan ve kuyruğu xs olan listeyi gösterir.

$$[x | xs] \equiv .(x, xs)$$

$$[x_1, \dots, x_n] \equiv .(x_1, \dots .(x_n, []))$$

- PROLOG'daki bir string, her bir tam sayının (integer) bir karakteri temsil ettiği tam sayıların bir listesidir.
- "hey" dizesi (string'i), 3 integer'dan oluşan bir listeyi belirtir.
- PROLOG, dinamik tipli bir dildir. Her türden değerler (sayılar, atomlar ve yapılar) birbirinin yerine kullanılabilir. Böylece farklı türlerdeki değerleri içeren heterojen listeler oluşturabiliriz.

PROLOG - Listeler

```
elemani(X, [X | Xs]).  
elemani(Y, [X | Xs]) :- elemani(Y, Xs).
```

```
sona_ekle([], Y, [Y]).  
sona_ekle([X|Xs], Y, [X|Ys]) :- sona_ekle(Xs, Y, Ys).  
?- sona_ekle([1, 2], 3, L).
```

```
birlestir([], L2, L2).  
birlestir([X|Xs], L2, [X|Ys]) :- birlestir(Xs, L2, Ys).  
?- birlestir([1, 2], [3, 4], L).
```


PROLOG - Geri İzleme (Backtracking)

```
female(marge).
female(lisa).
female(maggie).
female(wilma).
female(pebbles).
male(homer).
male(bart).
male(fred).
parent(marge, bart).
parent(homer, bart).
parent(marge, lisa).
parent(homer, lisa).
parent(marge, maggie).
parent(homer, maggie).
parent(wilma, pebbles).
parent(fred, pebbles).
father(X, Y) :-
    parent(X, Y),
    male(X).
```

GOAL father(X, bart).

Examine the next part of the rule

Y = bart

OUTPUT

PROLOG - Kesme (Cut)

- Bir telefon rehberindeki her kaydın şu şekilde bir yapısı olsun:

`kisi(Ad, Numara).`

- Bu rehberde adı aynı olan 2 kişi olmasın.

- Aşağıdaki önermeler 3'lü bir **ara** ilişkisi tanımlar:

`ara(Ad, Numara, [kisi(Ad, Numara) | Gerisi]).`

`ara(Ad, Numara, [_ | Gerisi]) :- ara(Ad, Numara, Gerisi).`

- Rehber şu liste olsun `[kisi(ali, 123), kisi(veli, 234), kisi(ayse, 123)]`

- Aşağıdaki sorgu çalıştırıldığında `Tel = 234` bulunur.

`?- ara(veli, Tel, Rehber).`

- Sonuç doğrudur ancak Prolog bu sonucu bulurken bütün rehberi tarar.

PROLOG - Kesme (Cut)

- Eğer bir sonuca ulaştığımızda bu sonucun yeterli olduğunu düşünüyorsak, Prolog'un tarama işlemini yarıda kesebiliriz.
- Prolog'da kesme sembolü ünlem işaretidir (!).
- ! başarılıdır (True).

```
ara_2(Ad, Numara, [kisi(Ad, Numara) | Gerisi]) :- !.
```

```
ara_2(Ad, Numara, [_ | Gerisi]) :- !, ara_2(Ad, Numara, Gerisi).
```

```
?- ara_2(X, 123, [kisi(ali, 123), kisi(veli, 234), kisi(ayse, 123)]).
```

```
X = ali.
```