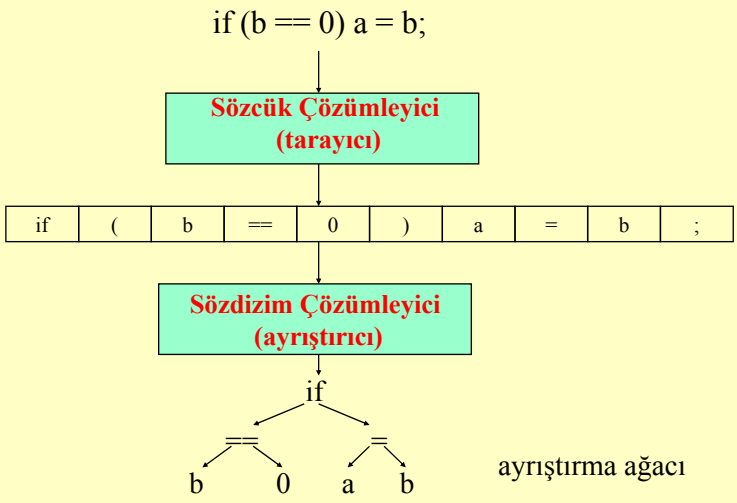
SÖZDİZİM ÇÖZÜMLEME

1

Sözdizim Çözümlemenin Zamanı

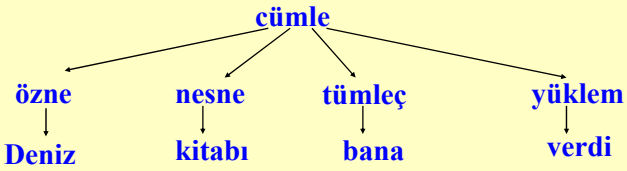


ayrıştırma ağacı

2

Ayrıştırma Örneği

- Doğal dillerde sözdizim çözümlemesi
  - Cümle yapısının dilbilgisi kurallarına uygunluğunu araştır
  - Her sözcüğün görevini belirle



"Deniz kitabı bana verdi"

3

Sözdizim Çözümlemenin Amacı

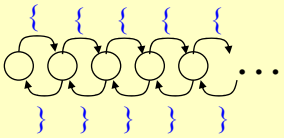
- Amaç– Kaynak programı oluşturan sözcüklerin programlama dilinin gramer kurallarına uygun bir sıralamada olup olmadıklarını belirleme
- Bu amaca ulaşmak için neye gerek duyulur?
  - Dilin gramer kurallarını (sentaksını) kesin ve açık bir şekilde tanımlamak için bir notasyon
  - Giriş sözcük katarının dilin sentaksına uygunluğunu belirlemek için bir yöntem
  - Bu gereksinimler karşılanırsa, herhangi bir dil için bir ayrıştırıcı otomatik olarak üretilebilir

4

Düzgün İfadeler Kullanılabilir Mi?

- Düzgün ifadeler sozcükleri tanımlamak için kullanılıyor ve DFA ile kolayca gerçeklenebiliyor. O halde dilin sentaksını belirlemek için de kullanılabilir mi?
  - HAYIR! – karmaşık kuralların tanımında yetersiz kalırlar
  - Örnek – İç içe yapılar (bloklar, ifadeler, deyimler) için dengeli parantezler:

{{} {} {{} { }}}



5

Bağlamdan Bağımsız Gramerler (BBG)

- 4 bileşenden oluşurlar:
  - Terminal simgeler (uç sözcükler) = sözcük veya ε
  - Non-terminal simgeler = sözdizimsel sınıf veya değişken
  - Başlangıç simgesi S = özel bir non-terminal
  - SOL→SAĞ şeklindeki türetim kuralları
    - SOL = bir tek non-terminal
    - SAĞ = terminal veya non-terminal katarı
    - Non-terminallerin açılımlarını tanımlar

S → a S a
S → T
T → b T b
T → ε

- Gramerin ürettiği dil başlangıç simgesinden yola çıkıp türetim kurallarını tekrar tekrar uygulayarak elde edilen sözcük katarları kümesidir
  - L(G) = G gramerinin ürettiği dil

6

BBG - Örnek

- Dengeli parantezler dili için gramer
  - S → (S) S
  - S → ε
    - 1 adet non-terminal: S
    - 2 adet terminal: "(", ")"
    - Başlangıç simgesi: S
    - 2 adet türetim kuralı
- Eğer bir sözcük katarı, dilin türetim kuralları uygulanarak elde edilebiliyor ise, o katar dilin gramer kurallarına (sentaksına) uygun yapıdadır
  - Örnek katar - "(())"
  - S = (S) S = (S) ε = ((S) S) ε = ((S) ε) ε =((ε) ε) ε = (())

7

BBG Gösterimi

- Nonterminaller genellikle şöyle gösterilir:
  - Küçük harfle isimler – deyim, ifade, vs.
  - Büyük harfler
  - S harfi – gramerin başlangıç simgesi
- Terminaller genellikle şöyle gösterilir:
  - Küçük harfler – a,b,x, vs
  - Özel işaretler – +, *, (,), 2,5, vs.
  - Özel sözcükler – if, while, for, vs.
- Yunan alfabesinden harfler, (α, β,γ, vs.) terminal ve/veya nonterminallerden oluşan bir gramer simgeleri katarını gösterir
  - A→ α

8

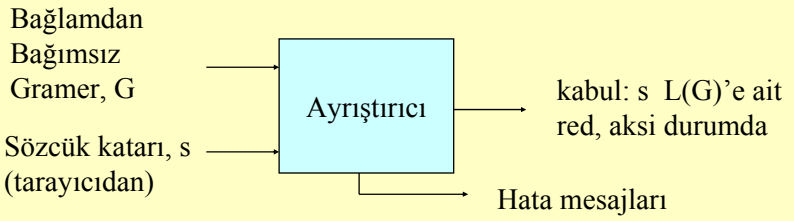
BBG Gösterimi (2)

- $A \rightarrow \alpha_1, A \rightarrow \alpha_2, \ldots, A \rightarrow \alpha_k$
  - A'nın birden fazla açılımı var ise, şöyle gösterilir

  $A \rightarrow \alpha_1 \mid \alpha_2 \mid, \ldots, \mid \alpha_k$
  - "α_i" A nonterminalinin açılım "seçenekleridir"
- Aksi belirtilmedikçe, ilk türetimde tanımlanan nonterminal gramerin başlangıç simgesidir
- Türetim işlemi – Başlangıç simgesinden başlayarak türetim kurallarının ard arda uygulanması işlemi
- Kabul – Verilen sözcük katarını üreten bir türetim işlemleri dizisinin varlığının belirlenmesi

9

Ayrıştırıcı



Sözdizim Çözümleyici (ayrıştırıcı)

•Bir G gramerini ve "s" sözcük katarını giriş bilgisi olarak alır, ve eğer "s" G'nin tanımladığı dile ait bir katar ise, bu katarı üreten türetim adımlarını verir.

•Aksi durumda, hata mesajları (sentaks hataları) üretir

10

Dİ, BBG'nin bir altkümesidir

Her Dİ (düzgün ifade) için bir G grameri tanımlanabilir

ε	S → ε
a	S → a
R1 R2	S → S1 S2
R1 | R2	S → S1 | S2
R1*	S → S1 S | ε

G1 = R1 için tanımlanan, S1 başlangıç simgesine sahip gramer
G2 = R2 için tanımlanan, S2 başlangıç simgesine sahip gramer

11

Örnek gramer: aritmetik ifade (toplama)

- Gramer
  - S → E + S | E
  - E → sayı | (S)
- Genişletilmiş gramer
  - S → E + S
  - S → E
  - E → sayı
  - E → (S)

4 türetim kuralı
2 non-terminal simge (S,E)
4 terminal simge: "(", ")", "+", sayı
başlangıç simgesi: S

12

Türetim İşlemi

- S başlangıç simgesiyle başla
- Türetim kurallarını kullanarak sözcük katarını türet
- α, β, γ gramer simgeleri katarları ve bir A → β türetim kuralı var ise, bir türetim adımı şöyle olacaktır:
  - o $\alpha A \gamma \Rightarrow \alpha \beta \gamma$ (A yerine β yerleştir)
  - o $S \overset{+}{\Rightarrow} \alpha$ ise, α G gramerinin bir cümlesidir

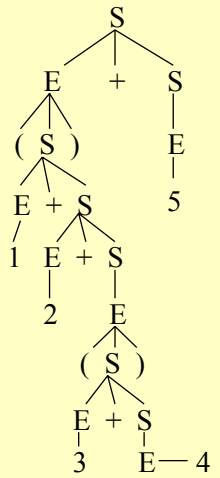
$\Rightarrow$ 1 adımda

$\overset{+}{\Rightarrow}$ 1/daha fazla adımda

- Örnek
  - •türetim kuralı: S → E + S
  - •(S + E) + E → (E + S + E) + E

  [S nonterminalini (S) açılımıyla (E + S) yer değiştir]

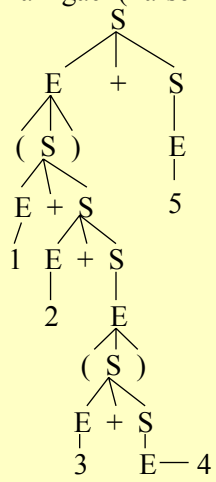
Ayrıştırma Ağacı



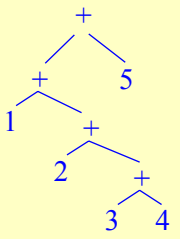
- •Ayrıştırma Ağacı = Türetim adımlarının bir ağaç yapısı üzerinde gösterilimi
- • Ara düğümlerde non-terminal simgeler
- • Yapraklarda terminal simgeler (sözcük)
- • Ancak türetim adımlarının sırası hakkında bilgi içermez

Ayrıştırma Ağacı / Soyut Sentaks Ağacı

Ayrıştırma Ağacı (Parse Tree)



Soyut Sentaks Ağacı (Abstract Parse Tree)



SSA içinde gereksiz bilgiler taşınmaz

15

Türetim Sırası

- Türetme adımı: türetim kuralları herhangi bir sırada seçilebilir, nonterminal simge yerine açılımı (türetim kuralının sağ tarafı) yerleştirilir
- İki standart türetim sırası vardır:
  - En-sağdan türetim
  - En-soldan türetim
- En-soldan türetim
  - Katar içinde en solda yer alan nonterminal simgeyi belirle ve bir türetim kuralı uygula
  - E + S → 1 + S
- En-sağdan türetim
  - Benzer şekilde ancak en sağda yer alan nonterminal simgeyi seç
  - E + S → E + E + S

16

Türetim Örnekleri

- **S → E + S | E**
- **E → sayı | (S)**
- En-soldan türetim: (1 + 2 + (3 + 4)) + 5

  S → E + S → (S)+S → (E+S) + S → (1+S)+S → (1+E+S)+S → (1+2+S)+S → (1+2+E)+S → (1+2+(S))+S → (1+2+(E+S))+S → (1+2+(3+S))+S → (1+2+(3+E))+S → (1+2+(3+4))+S → (1+2+(3+4))+E → (1+2+(3+4))+5
- En-sağdan türetim: (1 + 2 + (3 + 4)) + 5

  S → E+S → E+E → E+5 → (S)+5 → (E+S)+5 → (E+E+S)+5 → (E+E+E)+5 → (E+E+(S))+5 → (E+E+(E+S))+5 → (E+E+(E+E))+5 → (E+E+(E+4))+5 → (E+E+(3+4))+5 → (E+2+(3+4))+5 → (1+2+(3+4))+5

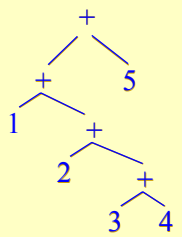
Sonuç: türetim sırası farklı ancak aynı ayrıştırma ağacı elde edildi

17

Belirsiz Gramerler

- Aritmetik toplama ifadesi için, en-sağdan ve en-soldan türetimler aynı ayrıştırma ağacını oluşturmuştur
- Türetim sırasından bağımsız olarak, + operatörü ayrıştırma ağacında sağ taraf ile ilişkilendirilir

(1+2+(3+4))+5 



18

Belirsiz Gramerler (2)

- + sağ taraf ile ilişkilendirilir çünkü türetim kuralı sağ-rekürsif özelliğe sahiptir: S → E + S
- Başka bir örnek gramer
  - S → S + S | S * S | sayı
  - Hem sağ-rekürsif, hem sol-rekürsif
- Belirsiz gramer = aynı cümlenin farklı türetimleri farklı ayrıştırma ağaçları oluştururlar
  - Eğer,aynı cümle için, iki farklı en-sağdan/en-soldan türetim var ise, G belirsizdir

19

Belirsiz Gramer - Örnek

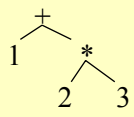
S → S + S | S * S | sayı

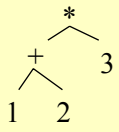
Aritmetik ifade: 1 + 2 * 3

Türetim 1: S → S+S → 1+S → 1+S*S → 1+2*S → 1+2*3

Türetim 2: S → S*S → S+S*S → 1+S*S → 1+2*S → 1+2*3

2 en-soldan türetim





Ancak, farklı ayrıştırma ağaçları!

20

Belirsizliğin Etkisi

- Farklı ayrıştırma ağaçları farklı değerler üretirler!
  Sonuç: programın anlamında tanımsızlık

Bir ayrıştırıcı ancak "**belirgin bir gramer**" (**belirsiz olmayan**) ile oluşturulabilir!

21

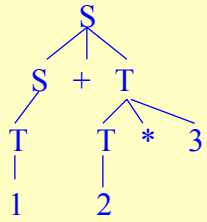
Belirsizlikten Kurtulma

- Belirsizlikten kurtulmak için, yeni nonterminal simgeler eklenir ve sadece sol-rekürsif veya sağ-rekürsif yapıya izin verilir

S → S + S | S * S | sayı

S → S + T | T

T → T * sayı | sayı



- T non-terminali işlem önceliğini belirler
- Sol-rekürsif yapı; sol taraf ile ilişkilendirme

22

Belirsizlikten Kurtulma (2)

- İşlem önceliğinin belirlenmesi
  - Her öncelik düzeyi için ayrı bir nonterminal tanımlanır
    - T nonterminal simgesi
  - Verilen bir öncelik düzeyinin operatörleri türetim kuralının sağ tarafı olarak belirlenirler
    - T → T * sayı
  - Yüksek öncelikli operatörlere erişim bir önceki düzeydeki öncelik non-terminali üzerinden gerçekleşir

İlişkilendirilme

- Bir operatör sol/sağda yer alan operand ile ilişkilidir veya bu tür ilişki yoktur (yok)
  - Sol: a + b + c = (a + b) + c
  - Sağ: a ^ b ^ c = a ^ (b ^ c)
  - Yok: a < b < c tanımsızdır
- Rekürsif özelliğin operatöre göre konumu ilişkilendirilme özelliğini belirler
  - Sol-rekürsif yapı; sol taraf ile ilişkilendirme
  - Sağ-rekürsif yapı; sağ taraf ile ilişkilendirme
  - yok: Rekürsif yapıyı kullanma. Operatörün her iki tarafında bir yüksek öncelik düzeyine ait non-terminal simgesine yer ver