

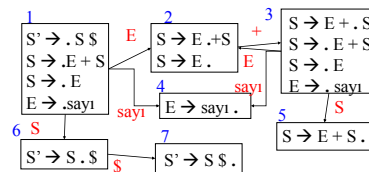
## SÖZDİZİM ÇÖZÜMLEME (5)

1

## Örnek Gramer İçin Uygulama

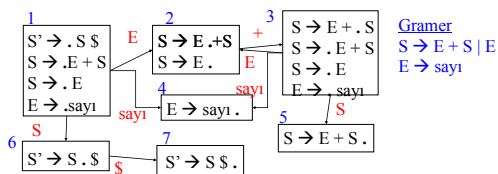
Aşağıdaki gramer için oluşturulan DFA

$S \rightarrow E + S \mid E$   
 $E \rightarrow \text{sayı}$



2

## LR(0) Ayrıştırma Tablosu



Durum 2:  
ötele/indirge  
çelişkisi?

	sayı	+	\$	E	S
1	ö4			2	6
2	S → E	ö3/S → E	S → E		

3

## Çelişkinin Giderilmesi

- Çelişkinin giderilmesi için sonraki sözcükten yararlanılır. Bir adet sonraki sözcük kullanan üç farklı yöntem vardır:
  - SLR (Simple LR)
  - LALR (LookAhead LR)
  - LR(1)
- Her biri sonraki sözcükten farklı şekilde yararlanır ve farklı işlem yetenekleri elde eder

4

## SLR Ayırıştırıcı

- SLR Ayırıştırıcı = LR(0)'ın bir uzantısı
  - Her  $X \rightarrow \beta$  türetim kuralı için, sonraki sözcük "c"e bak
  - İndirgeme işlemini yalnızca "c" İZLE(X) kümesinde ise uygula
- SLR ayrıştırma tablosu
  - İndirgeme satırları dışında LR(0) tablosu ile aynıdır
  - $X \rightarrow \beta$  indirgeme işlemini sadece İZLE(X) kümesi elemanlarına ait sütunlara ekle

Örnek: İZLE(S) = {S}

Grameri  
 $S \rightarrow E + S \mid E$   
 $E \rightarrow \text{sayı}$

	sayı	+	\$	E	S
1	ö4			2	6
2	<del>S</del> → E	ö3/ <del>S</del> → E	S → E		

5

## SLR Ayırıştırma Tablosu

- İndirgeme işlemleri tüm satırı kaplamaz
- Tablonun diğer bölümleri LR(0) tablosu ile aynıdır

Grameri

$S \rightarrow E + S \mid E$

$E \rightarrow \text{sayı}$

	sayı	+	\$	E	S
1	ö4			2	6
2		ö3	S → E		
3	ö4			2	5
4		E → sayı	E → sayı		
5		S → E + S			
6		ö7			
7		kabul			

6

## SLR'nin yetersiz olma durumu:

Grameri

$S \rightarrow L = R$

$S \rightarrow R$

$L \rightarrow *R$

$L \rightarrow d$

$R \rightarrow L$

L: bir atama deyiminde sol taraf

R: bir atama deyiminde sağ taraf

\*: işaretçi referansı

SLR(1) için DFA durumları hesaplandığı zaman, aşağıdaki durum oluşacaktır:

$S \rightarrow L = R$
$R \rightarrow L =$

Çelişki var mıdır? EYET

7

## SLR'nin yetersiz olma durumu:

$I_0: S' \rightarrow S$        $I_1: S' \rightarrow S \cdot$        $I_2: S \rightarrow L = R \dots$        $I_6: S \rightarrow L = \cdot R \dots$

$S \rightarrow \cdot L = R$

$S \rightarrow \cdot R$

$L \rightarrow \cdot *R$

$L \rightarrow \cdot d$

$R \rightarrow \cdot L$

İZLE(R) = {=}

Grameri

$S \rightarrow L = R$

$S \rightarrow R$

$L \rightarrow *R$

$L \rightarrow d$

$R \rightarrow L$

$I_2$  durumunu ele alalım

$S \rightarrow L = \cdot R$  Geçiş(2,=) = 6 → tablo girişi "ötele 6"

$R \rightarrow L \cdot$  → tablo girişi "indirge R → L"

SONUÇ: ötele/indirge çelişkisi

8

## LR(1) Ayırıştırıcısı

- LR(1) gramer = ötele/indirge ayırıştırıcısı tarafından **bir adet sonraki sözcük** kullanarak tanınabilen gramer
- LR(1) ayırıştırıcısı LR(0)'a benzer kavramlara dayanır
  - Ayırıştırıcı durumları = parçalar kümesi
  - LR(1) parçası = LR(0) parçası + **türetimi izlemesi olası bir adet sonraki sözcük**
    - LR(0) parçası:  $S \rightarrow . S + E$
    - LR(1) parçası:  $S \rightarrow . S + E, +$
    - Sonraki sözcük sadece İNDİRGEME işlemleri üzerinde etkindir.
    - İndirgeme işlemi sadece giriş simgesinin sonraki sözcük ile aynı olma durumunda uygulanır

9

## LR(1) Durumları

- LR(1) durumu = LR(1) parçaları kümesi
- LR(1) parçası =  $(X \rightarrow \alpha . \beta, y)$ 
  - Anlamı:  $\alpha$  halen yığının en üstünde yer alan simgelerle eşleştirilmiştir, bir sonraki aşamada  $\beta$  y'nin görülmesi umulmaktadır
- Kestirme İfade Şekli
 

$S \rightarrow S . + E$	$+, S$
$S \rightarrow S + . E$	$\text{sayı}$

  - $(X \rightarrow \alpha . \beta, x1)$
  - ...
  - $(X \rightarrow \alpha . \beta, xn)$  yerine  $(X \rightarrow \alpha . \beta, \{x1, \dots, xn\})$
- Kılıf ve Geçiş işlemlerinin uygun şekilde geliştirilmesi gerekir

10

## LR(1) Kılıf İşlemi

- LR(1) kılıf işlemi:
  - Başlangıçta  $\text{Kılıf}(S) = S$  atamasını yap
  - $S'$ 'in içerdiği her  $X \rightarrow \alpha . Y \beta, z$  parçası ve her  $Y \rightarrow \gamma$  türetimi için, aşağıdaki parçayı  $\text{Kılıf}(S)$  kümesine ekle  $Y \rightarrow . \gamma, \text{ILK}(\beta z)$
  - Yukarıdaki adımı kümeye yeni parça eklenemez olana kadar tekrarla
- LR(0) Kılıf işlemine benzer fakat sonraki sözcükler de gözönüne alınmaktadır

11

## LR(1) Başlangıç Durumu

- Başlangıç durumu:  $(S' \rightarrow . S, \$)$  parçası ile başla ve kılıf işlemi uygula

- Örnek:

$S' \rightarrow S S$   
 $S \rightarrow E + S \mid E$   
 $E \rightarrow \text{sayı}$

$X \rightarrow \alpha . Y \beta, z$  parçası ve her  $Y \rightarrow \gamma$   
 $Y \rightarrow . \gamma, \text{ILK}(\beta z)$  parçasını ekle

$S' \rightarrow . S, \$$



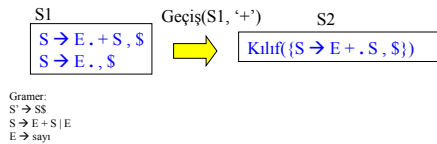
$\text{ILK}(S) = \{\text{sayı}, (\}$   
 $\text{ILK}(S') = \{\epsilon, +\}$   
 $\text{ILK}(E) = \{\text{sayı}, (\}$

$S' \rightarrow . S, \$$   
 $S \rightarrow . E + S, \$$   
 $S \rightarrow . E, \$$   
 $E \rightarrow . \text{sayı}, \$$   
 $E \rightarrow . \text{sayı}, +$

12

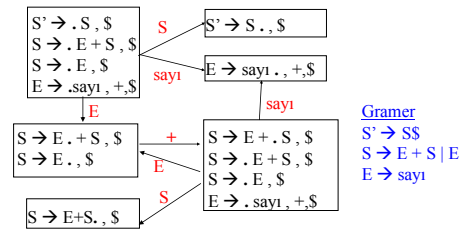
## LR(1) Geçiş İşlemi

- LR(1) geçiş işlemi LR(1) durumları arasındaki geçişleri tanımlar
- Algoritma: Bir S durumu ve Y simgesi için
  - Eğer  $[X \rightarrow \alpha . Y \beta]$  parçası I içinde yer alıyor ise  
Geçiş(I, Y) = Kılıf( $[X \rightarrow \alpha Y . \beta]$ )



13

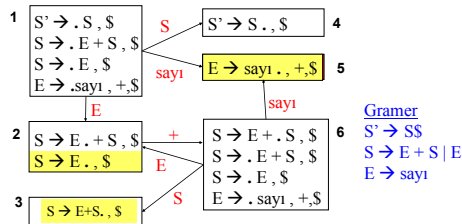
## Örnek Gramer İçin DFA



14

## LR(1) İndirgemeler

- İndirgemeler  $(X \rightarrow \gamma . y)$  şeklindeki LR(1) parçalarına karşı düşerler



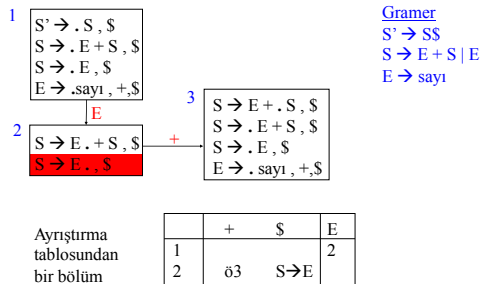
15

## LR(1) Ayrıştırma Tablosunun Oluşturulması

- İndirgeme işlemleri dışında, LR(0) ayrıştırma tablosu için yürütülen adımların aynısı
- x terminali üzerinden  $S \rightarrow S'$  durum geçişi için:
  - $\text{tablo}[S, x] = \text{ötele}(S')$  [ $S' = \text{geçiş}(S, x)$ ]
- N non-terminali üzerinden  $S \rightarrow S'$  durum geçişi için:
  - $\text{tablo}[S, N] = (S')$  [ $S' = \text{geçiş}(S, N)$ ]
- Eğer I  $\{(X \rightarrow \gamma . y)\}$  parçasını içeriyor ise:
  - $\text{tablo}[I, y] = \text{indirge}(X \rightarrow \gamma)$  (sadece y sütununa ekle)

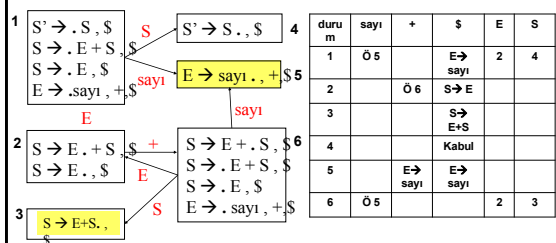
16

## Örnek LR(1) Ayırıştırma Tablosu



17

## LR(1) Ayırıştırma Tablosu



18

## ANLAMSAZ ÇÖZÜMLEME

(Semantic Analysis)

19

## Anlamsal Çözümleme

- Tarama ve ayırıştırma yöntemleri değişkenlerin, nesnelerin veya fonksiyonların **doğru kullanımını** denetleyecek güce sahip değildirler
- Sözcük ve sözdizim çözümlemesi başarıyla tamamlanmış program kodunda hala **anlamsal hatalar** yer alıyor olabilir
- Anlamsal Çözümleme:** Programlama yapılarının (değişkenler, ifadeler, deyimler, fonksiyonlar, vs.) kullanımına ilişkin belirli kurallara uyulduğunu belirlemekle yükümlüdür

20

## Anlamsal Çözümleme Sınıfları

- Örnek anlamsal kurallar
  - Değişkenler kullanılmadan önce tanımlanmış olmalı
  - Bir değişken birden fazla kez tanımlanamaz
  - Bir atama deyiminde, değer atanan değişken ve hesaplanan ifade değeri aynı tipten olmalıdır
  - Bir if deyiminde, kontrol ifadesi lojik tipten sonuç üretmelidir
- Anlamsal kurallar iki sınıfa ayrılır:
  - **Tiplere** yönelik anlamsal kurallar
  - **Bağlama** yönelik anlamsal kurallar

21

## Tip Bilgisi ve Tip Kontrolü

- **Tip Bilgisi:** Farklı yapıların (değişken, deyim, ifade, vs) hangi tipte ilişkilendirileceğini belirler
  - değişken: `int a;` integer
  - ifade: `(a+1) == 2` boolean
  - deyim: `a = 1.0;` floating-point
  - fonksiyon: `int pow(int n, int m)` `int = int, int`
- **Tip Kontrolü:** Programın farklı yapılarının tip tutarlılığını sağlayan kurallar kümesi

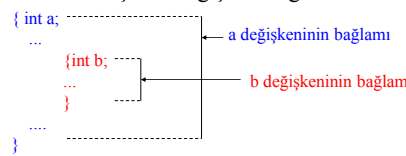
22

## Bağlam Bilgisi (Scope)

- Bağlam bilgisi: Değişken adlarının (değişken, fonksiyon adı, etiket, vs.) **bildirimi** ve **kullanımlarına** izin verilen program bölümlerini tanımlar
- Sözcüksel Bağlam: Kod içindeki metin bölümü (lexical scope)
  - Örnek: deyim bloğu, parametre listesi, fonksiyon kodu, programın tümü
- Bir değişkenin bağlamı: **Değişken bildirimi ile belirlenen sözcüksel bağlamdır**

23

## Değişken Bağlamı

- Deyim blokları içinde değişken bağlamı:
 
- Global değişkenlerin bağlamı: kaynak dosya
- Dış (external) değişkenlerin bağlamı: tüm program

24

## Parametre ve Etiket Bağlamı

- Fonksiyonun formel parametreleri bağlamı:

```
int fonk (int n){
  ...
}
```

← n parametresinin bağlamı

- Etiket bağlamı:

```
void fonk() {
  ... goto e1;
  ...
  e1: i++;
  ...
  ... goto e1;
}
```

← e1 etiketinin bağlamı (tüm fonksiyon)

25

## Bağlama İlişkin Anlamsal Kurallar

- Kurallar:

- Kural 1: Her değişken sadece kendine ait bağlam içinde kullanılmalıdır
- Kural 2: Aynı sözcüksel bağlam içinde, aynı tipten ve aynı değişken adına sahip değişkenler birden fazla kez tanımlanamaz

```
int X(int X) {
  int X;
  goto X;
  {
    int X;
    X: X = 1;
  }
}
```

Hatalı bildirimler

26

## Sembol Tablosu (Symbol Table)

- Anlamsal kontroller program değişkenlerinin bağlam ve tiplerine yönelik işlemler gerektirir
- Değişkenlere ait bu bilgilerin saklanabileceği bir ortama gerek duyulur → **sembol tablosu**
- Sembol tablosunun her kaydı şu bilgileri içerir:
  - Değişkenin adı
  - Değişkene ait diğer bilgiler: türü, tipi, sabit, vs.

ADI	TÜR	TİP	ÖZELLİK
fonk	fonksiyon	int, int → int	dış değişken
m	arg	int	
n	arg	int	sabit
isim	değişken	char	sabit

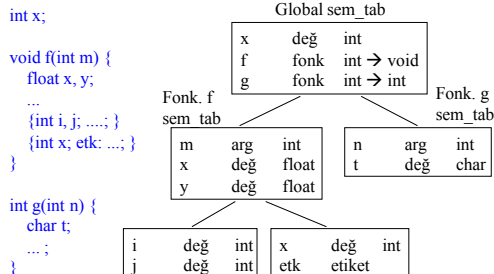
27

## Bağlam Bilgisi

- Bağlam bilgisi sembol tablosunda nasıl saklanır?
- Çözüm:
  - Program içinde bir bağlam hiyerarşisi vardır
  - Benzer bir sembol tabloları hiyerarşisi oluştur → her bağlam için bir tablo
  - Her sembol tablosu karşı düştüğü sözcüksel bağlam içinde tanımlı olan değişken adlarını içerecektir

28

## Örnek



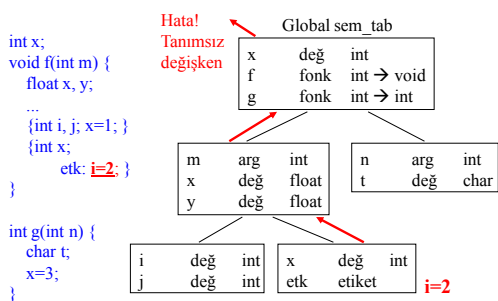
29

## Benzer Değişken Adları

- Sembol tablolarının hiyerarşik yapısı değişken adı çakışmalarını otomatik olarak çözümler
  - Çakışmaya örnek: aynı değişken adına sahip ve örtüşen bağlamlar içinde kullanılan değişkenler
- Bir program noktasında, bir değişkene ait bildirimin hangisi olduğunu bulmak için:
  - O anda geçerli olan bağlamdan başla,
  - Aynı adı taşıyan bir değişken bildirimi bulana kadar hiyerarşi üzerinde yukarı doğru ilerle

30

## Anlamsal Hataların Yakalanması



31

## Sembol Tablosu İşlemleri

- İki temel işlem
  - Ekleme** işlemi: Değişkenlere ait yeni kayıtların eklenmesi
  - Arama** işlemi: Tabloda yer alan bilgilere ulaşmak için sorgulama işlemi
- Sembol tablosu, tarayıcı tarafından sözcük çözümleme aşamasında oluşturulamaz, çünkü bağlam hiyerarşisi sözdizim içinde yer alır
- Sembol tablosu ayrıştırma işlemleri esnasında, “anlamsal işlemler” yürütülerek oluşturulur
- Hash tablosu kullanarak sembol tablosunu etkin bir şekilde gerçeklemek mümkündür

32



## Tip Kontrolü

- Tip nedir?
  - Programın yürütülmesi sırasında hesaplanan değerleri tanımlar
  - Hesaplanan değer üzerinde bir önermedir
    - Örnek: "int x"  $\rightarrow -2^{31} \leq x < 2^{31}$
- Tip Hatası: Yürütme sırasında ortaya çıkan çelişkili işlemler
- Amaç: tip hatalarından arındırılmış kod

33

## Tip Hataları Nasıl Engellenir

- İki adımda: Tip Ataması ve Tip Kontrolü
- **Tip ataması:** programdaki yapıların tiplerini tanımlar
  - Açık (int x) veya dolaylı (x=1) olabilir
  - **Tip tutarlığı:** tip atamalarına göre doğru kullanım
- **Tip Kontrolü:** programın tip atamalarını doğru kullanıp kullanmadığının denetimi
  - Bir dizi tip kontrolü kuralları içerir

34

## Tip Kontrolü

- Tip kontrolü için anlamsal işlemler
- Kurallara örnekler
  - Tekil ve ikili operatörler ( örnek: -, +, == ) uygun tipten operandlar almalı
  - Fonksiyonlar doğru sayıda ve tipten argümanlarla çağrılmalı
  - Return deyimi fonksiyon tipiyle uyumlu olmalı
  - Atama deyiminde, atanan değerın tipi, atamanın yapıldığı değişkenin tipiyle uyumlu olmalı

35