

# **Biçimsel Diller ve Otomata Teorisi**

*Dr. Öğr. Üyesi Hayri Volkan Agun  
Bilgisayar Mühendisliği Bölümü  
Bursa Teknik Üniversitesi*

# Kaynaklar

## Ders Kitabı

- An Introduction to Formal Languages and Automata, Peter Linz, 6th Edition, 2017.
- An Introduction to Computer Theory, Daniel Isaac Aryeh Cohen, 2nd Edition, 1996.

## İçerik

- %100 Teorik
- Klasik sınav
- Vize %40, Final %60

# Bağlam Bağımsız Diller

- $L=\{a^n b^n : n \geq 0\}$  dilinin aslında düzenli bir dil olmadığını gördük. Bu dil aslında bilgisayar programlamada, HTML ve XML yapılarında sıklıkla karşımıza çıkmaktadır. Örneğin; başlangıç etiketli DIV olan bir HTML sayfasının içerisinde bir çok iç içe DIV etiketi geçmektedir. Burada açılan DIV etiketi ile kapanan DIV etiketleri aynı sayıda olmak durumundadır. Bu dil yukarıdaki dil tanımına birebir uyar.
- Düzenli olmayan dil ailesi içinde bağlamdan bağımsız olarak tanımlanan bir dil vardır. Bu dillere Bağlam Bağımsız Diller denir.

# Bağşam Bağımsız Gramer

- Gramer dil için tanımlı kuralları ifade eder. Bir dililin bağımsız olması için

$$G = (V, T, S, P)$$

- V kural adlarını, T sözlüğü, S başlangıç ve P kuralları ifade eder. Burada tüm kurallar aşağıdaki gibi ise

$$A \rightarrow x,$$

- Burada A kural adını ve x sembolü ise ya bir kural adını yada bir sözlük elemanını temsil etmektedir.
- Burada yukarıdaki kural tanımlaması sağlanırsa bu gramer bağlam bağımsız gramer olarak adlandırılır.

# Grammer Örnek

- Aşağıda tanımlı gramer için dil tanımını yapınız?


$$\begin{aligned} S &\rightarrow abB, \\ A &\rightarrow aaBb, \\ B &\rightarrow bbAa, \\ A &\rightarrow \lambda, \end{aligned}$$

- Bu gramer S,A, ve B kural adlarına sahiptir. Açılımları yaparak ne tür bir yapıyı yakaladığını bulabiliriz.
- $L = \{ab((bb)(aa))^n(bba)(ba)^n : n \geq 0\}$

# Grammer Örnek

- $L=\{a^n b^m : n \neq m\}$  ile tanımlanan dil bağlam bağımsız mıdır?
- Bunu bulmak için bu dili ifade eden bağlam bağımsız bir gramer oluşturmamız yeterlidir?
- Örneğin  $n$  sayısı  $m$ 'den 1 fazla olacak şekilde bir gramer oluşturabiliriz. Önce eşit sayıda  $a$  ve  $b$  için bir gramer tanımlayıp sola bir fazla  $a$  ekleyebiliriz.

extra a eklenir



```

$$\begin{aligned} S &\rightarrow AS_1, \\ S_1 &\rightarrow aS_1b|\lambda, \\ A &\rightarrow aA|a. \end{aligned}$$

```

- Bu kurallar bağlam bağımsız gramer şartını sağlamaktadır. Dolayısıyla dil de bağlam bağımsızdır.

# Türetim

- Gramer kuralları her zaman soldaki öge üzerinden türetiliyorsa en soldan türetim, sağdaki ögeden türetiliyorsa en sağdan türetim grameri adını alırlar.

- Örneğin;

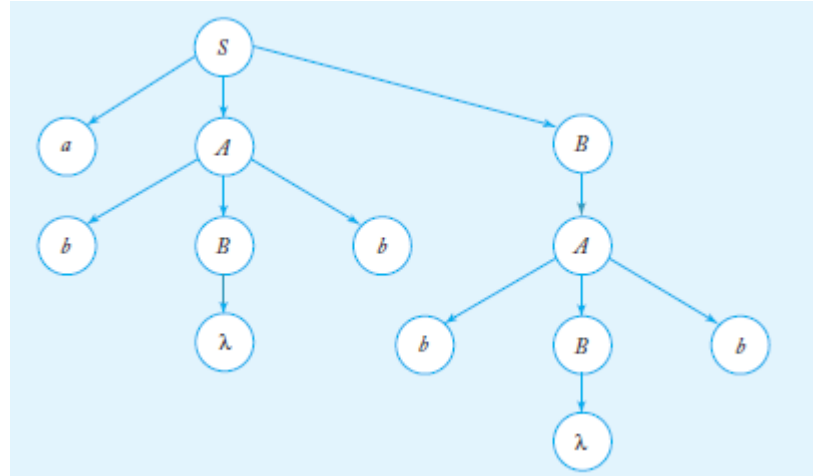
$$\begin{aligned} S &\rightarrow aAB, \\ A &\rightarrow bBb, \\ B &\rightarrow A|\lambda. \end{aligned}$$

- En soldan türetim :  $S \Rightarrow aAB \Rightarrow abBbB \Rightarrow abAbB \Rightarrow abbBbbB \Rightarrow abbbbB \Rightarrow abbbb$
- En sağdan türetim:  $S \Rightarrow aAB \Rightarrow aA \Rightarrow abBb \Rightarrow abAb \Rightarrow abbBbb \Rightarrow abbbb.$

# Ağaç Yapısı

- Aşağıda kuralları verilen gramer ile “abbbb” karakter katarı için türetim ağaç yapısı verilmiştir.

$S \rightarrow aAB,$   
 $A \rightarrow bBb,$   
 $B \rightarrow A|\lambda.$





# Ağaç Yapısı Soru

- Aşağıda kuralları verilen gramer ile  $w = abbbaabbaba$  karakter katarı için türetim ağaç yapısını çiziniz?

$S \rightarrow abB,$   
 $A \rightarrow aaBb,$   
 $B \rightarrow bbAa,$   
 $A \rightarrow \lambda,$

# Ayrıştırma (Parsing)

- Gramer kurallarını kullanarak verilen bir karakter katarı için ağaç yapısını bulma işine ayrıştırma denmektedir.
- Bazen gramer kuralları birden fazla kez veya farklı sırada aynı karakter katarı için uygulanabilir. bu durumda birden fazla farklı ağaç yapısı elde edilir.
- Bu şekilde birden fazla ağaç yapısı elde edilen durumlar için ayrıştırma işlemi belirsiz olur. Belirsizlik gramer kurallarından kaynaklanır.
- Çeşitli ayrıştırma yöntemleri vardır. Önceki sayfada uyguladığımız yöntem brute force yada yukarıdan aşağı ayrıştırma yöntemidir.

# Ayrıştırma (Parsing)

- Aşağıda gramer kuralları verilen dil ve  $w=aabb$  karakter katarı için ayrıştırma işlemini yaparak dilin bu karakter katarını kabul edip etmediğini bulunuz?

$$S \rightarrow SS \mid aSb \mid bSa \mid \lambda$$

- İlk adım için tüm gramer kurallarını yukarıdan aşağıya doğru açalım. Burada  $w$ 'nin ilk harfi  $a$  olduğu için 1. ve 2. kurallar çalışacaktır.

1.  $S \Rightarrow SS,$
2.  $S \Rightarrow aSb,$
3.  $S \Rightarrow bSa,$
4.  $S \Rightarrow \lambda.$

# Ayrıştırma (Parsing)

- Aşağıda gramer kuralları verilen dil ve  $w=aabb$  karakter katarı için ayrıştırma işlemini yaparak dilin bu karakter katarını kabul edip etmediğini bulunuz?
- İkinci adımda 1. ve 2. kuralları açarak devam edelim. Burada 1. kural için açılım yaparken ara formları aşağıdaki gibi oluşturabiliriz.

$$\begin{aligned} S &\Rightarrow SS \Rightarrow SSS, \\ S &\Rightarrow SS \Rightarrow aSbS, \\ S &\Rightarrow SS \Rightarrow bSaS, \\ S &\Rightarrow SS \Rightarrow S, \end{aligned}$$

# Ayrıştırma (Parsing)

- İkinci adımda yine 2. kural için açılımlar aşağıdaki gibi olacaktır.

$$\begin{aligned} S &\Rightarrow aSb \Rightarrow aSSb, \\ S &\Rightarrow aSb \Rightarrow aaSbb, \\ S &\Rightarrow aSb \Rightarrow abSab, \\ S &\Rightarrow aSb \Rightarrow ab. \end{aligned}$$

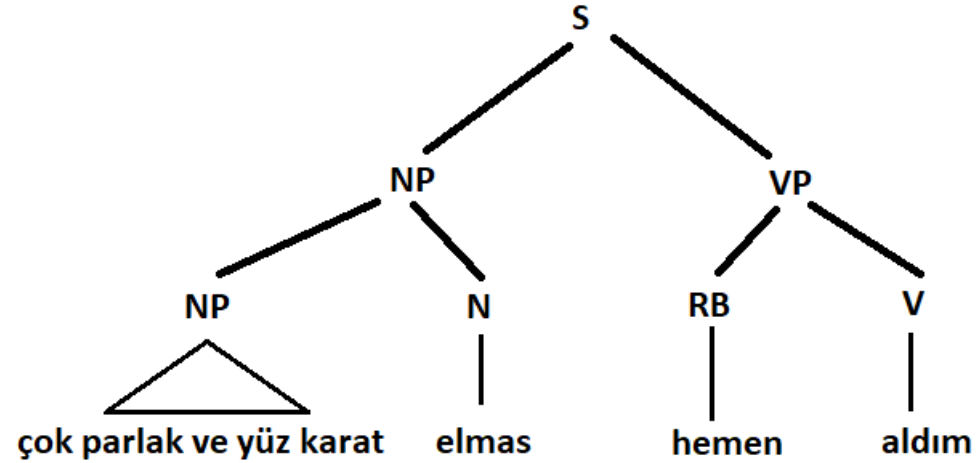
- Birinci ve ikinci kurallardan a ile başlamayanları eleyebiliriz. Bundan sonra (aa'dan sonra) 3. ve 4. karakter b olacaktır. Bu durumda 3. kural ve 4. kuralı eleyebiliriz. Şartları sağlayan sadece 2. açılım olacaktır. Burada aaSbb şartı sağlaması için S yerine  $\lambda$  (boş) açılım (4. gramer kuralı) olacaktır. Bu durumda bu dil grameri ile aabb karakter katarı kabul edilir.

# Ayrıştırma Belirsizlik

- Tek bir grammer ve tek bir cümle yada öbek için oluşan ağaç yapılarının sayısı ile ifade edilebilir.
- Örneğin: verilen bir gramer (kura dizisi) ve sözlük için ayrıştırma kuralları aşağıdaki gibi tanımlansın:
  - $S \rightarrow NP VP$      $NP \rightarrow ADJ NP$      $NP \rightarrow ADJ N$      $NP \rightarrow ADJ ADJ$      $VP \rightarrow RB VP$
  - $VP \rightarrow RB V$      $CP \rightarrow CONJ NP$      $NP \rightarrow NP CP$
  - Sözlük: [karat/N, çok/RB, parlak/ADJ, çok/ADJ, elmas/N, V/aldım, yüz/ADJ, yüz/V, RB/hemen]

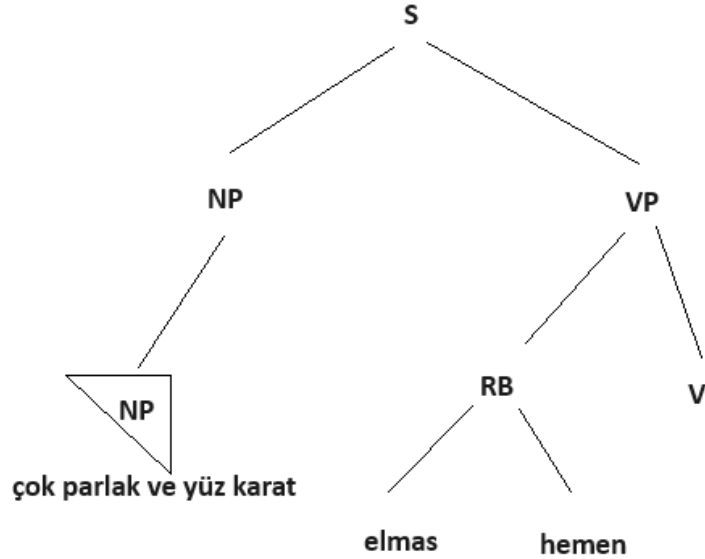
# Ayrıştırma Belirsizlik

- Bu gramer yapısını ve sözüğe göre birden fazla ağaç yapısı elde edebiliriz. Örneğin;
- Cümlesi: çok parlak ve yüz karat elmas hemen aldım
- Bu durumda ağaç yapısı sadece bir adet olur.



# Ayrıştırma Belirsizlik

- Peki kural yazpısını biraz değıştirelim. Ve N (elmas) ve RB (hemen) Adj olarak alabiliriz. Bu kuralları eklersek
- Kural: RB -> N Adj Bu durumda İkinci ağaç yapısı elde edilebilir.





# Belirsizlik

- Eğer bir grammar birden fazla ağaç yapısı üretiyor ise bu gramerle tanımlanan dil için kalıtsal belirsiz dil deriz.
- Örneğin aşağıdaki dil tanımı için verilen dil kalıtsal olarak belirsizdir.

$$L = \{a^n b^n c^m\} \cup \{a^n b^m c^m\},$$

- Örneğin L dili L1 ve L2 diye iki farklı kurallar verilsin.

$$\begin{aligned} S_1 &\rightarrow S_1 c | A, \\ A &\rightarrow a A b | \lambda \end{aligned}$$

- Sonuçta L1 ve L2 dil gramerlerini OR yada  $|$  kullanarak birleştiririz.

$$S \rightarrow S_1 | S_2.$$

# Ayrıştırma

- Soru aşağıda gramer kuralları verilen bir dil için L dil tanımını yapınız?

$$S \rightarrow aSB|bSA$$

$$S \rightarrow \lambda$$

$$A \rightarrow a$$

$$B \rightarrow b$$

- Önce bir açılım yapalım: 1. adımda kabul edilen kurallar  $aSB$  ve  $bSA$  'yı oluşturulur. Bu kurallardan ilkini açarsak  $a(aSB|bSA)b$  ifadesi bulunur. İkinci kural içinde  $b(aSB|bSA)a$  ifadesi bulunur.
- Açma işlemine devam edelim ancak burada bir ağaç yapısı oluşturmamız gerekir. Bu ağaç yapısında her bir kural açılımı bir düğüme denk gelmelidir.

# Ayrıştırma

- $a(aSB | bSA)b \Rightarrow a(a(aSB) | (b(bSA))) | (b((aSB) | (bSA)))$
- $b(aSB | bSA)a \Rightarrow b((a(aSB)) | (b(aSB))a) | (b(bSA)a) | (b(aSB)a)$
- Burada S yi turetmeyi durduralım ( $S \rightarrow \lambda$ ) ve A ve B sembollerini türetelim.
- 1. Kural için son değerler :  $aab | bba | bab | ba$
- 2.Kural için son değerler:  $baab | baba | bbaa | baba$
- $L = \{(a^n b^m)^k : n \geq 0, m \geq 0, k > 0\}$  \* Doğruluğunu çelişki yöntemi ile ispatlayınız?

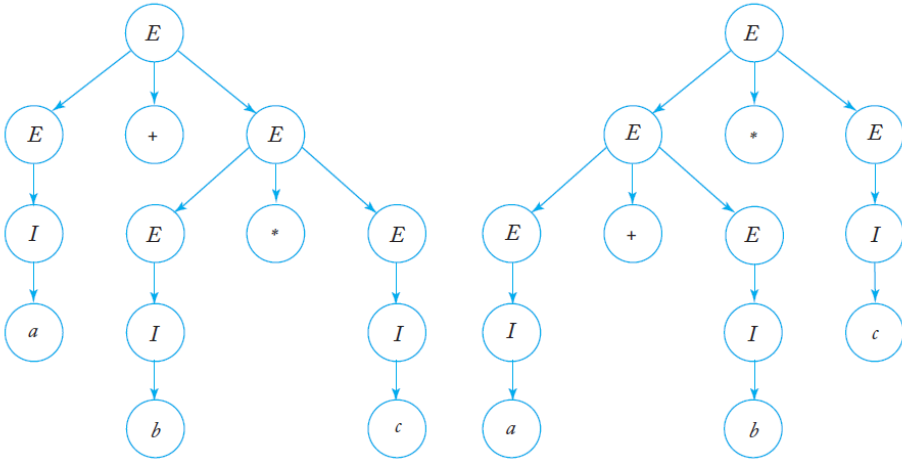
# Ayrıştırma

- C programlama dili gibi dillerde hesaplama yapmak için aşağıdaki gibi bir gramer tanımlanmış olsun. Bu durumda bu gramer kuralları ile kabul edilen dil  $a + b * c$  işlemi için doğru çalışır mı? Yada diğer bir deyişle belirsizlik oluşturmaz mı?

$$\begin{aligned} E &\rightarrow I, \\ E &\rightarrow E + E, \\ E &\rightarrow E * E, \\ E &\rightarrow (E), \\ I &\rightarrow a | b | c. \end{aligned}$$

- Hayır.  $E \rightarrow E + E$  açılımı ile başlayalım. Bu  $a + b$  ifadesini  $E \rightarrow I$  kuralını kullanarak kabul edecektir.  $E \rightarrow E * E$  ve  $I \rightarrow c$  kuralı ile de  $(a + b) * c$  kabul edilir. Ancak bu ifade doğru bir dört işlem olmaz. Çünkü  $*$  işleminin önceliği daha fazladır.

# Belirsizlik



- $(a + b) * c$  ifadesinin verilen gramer kuralları kullanılarak oluşturduğu ağaç yapıları yanda verilmiştir.
- Bu ağaç yapılarından sağdaki doğru bir 4 işleme karşılık gelmektedir. Çünkü parantez önceliği hesaba katılmıştır.
- Gramer kurallarını değiştirerek bu belirsizliği giderebiliriz.

# Belirsizlik

- Yeni gramer kuralları eklersek ve ifadeyi ayrıştırırsak yandaki ağaç yapısını elde edebiliriz.

$$E \rightarrow T,$$

$$T \rightarrow F,$$

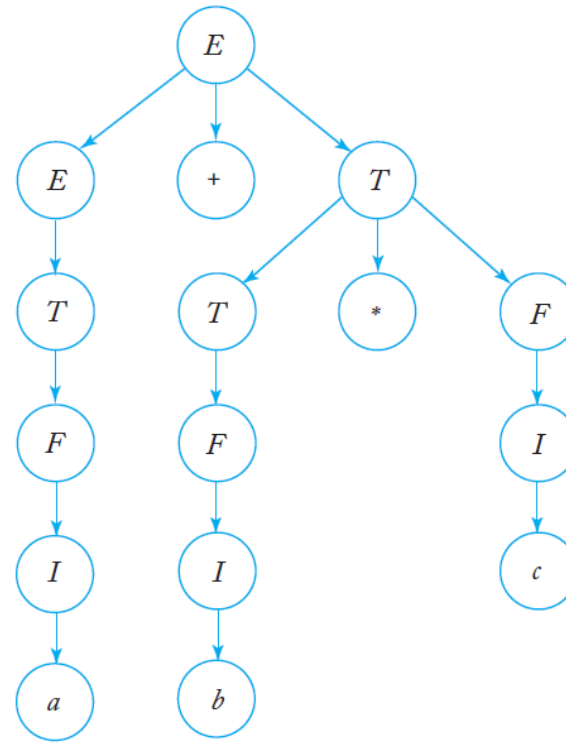
$$F \rightarrow I,$$

$$E \rightarrow E + T,$$

$$T \rightarrow T * F,$$

$$F \rightarrow (E),$$

$$I \rightarrow a|b|c.$$



# Ayrıştırma

- Aşağıda verilen dil tanımları için gramer kurallarını çıkartınız?

(a)  $L = a^n b^n$ ,  $n$  is even.

(b)  $L = a^n b^n$ ,  $n$  is odd.

(c)  $L = a^n b^n$ ,  $n$  is a multiple of three.

(d)  $L = a^n b^n$ ,  $n$  is not a multiple of three.