FORMAL DİLLER VE OTOMATLAR

BAĞLAMDAN BAĞIMSIZ DİLBİLGİSİ (CONTEXT-FREE GRAMMARS)(CFG)

Bir (V,T,S,P) gramerinde tüm türetim kuralları (P) , A $\in$ V ve x $\in$ (V U T)* olmak üzere A → x şeklinde ise bağlamdan bağımsızdır. Yani bağlamdan bağımsız her gramer kuralının sol tarafında tek bir değişken vardır.

Bu koşulu karşılamayan gramer kuralları olabilir. Örneğin 1Z1→ 101 kuralında Z sadece sağında ve solunda 1 olması durumunda 0 olur. Bu tür gramer yapılarına context-sensitive (bağlama duyarlı) denmektedir.

$$\left.\begin{array}{l} A \rightarrow x_1 \\ A \rightarrow x_2 \\ A \rightarrow x_3 \\ : \\ A \rightarrow x_k \end{array}\right\}$$

şeklindeki türetim kuralları $A \rightarrow x_1 \mid x_2 \mid \ldots.. \mid x_k$ şeklinde yazılabilir.

Örneğin; $E \rightarrow E+E \mid E*E \mid (E) \mid id$ türetim kuralları ile verilmiş bir G CFG'sinde;

$$
\begin{aligned}
E &\Rightarrow E*E \\
&\Rightarrow (E)*E \\
&\Rightarrow (E)*id \\
&\Rightarrow (E+E)*id \\
&\Rightarrow (E+id)*id \\
&\Rightarrow (id+id)*id
\end{aligned}
$$

Türetim

$\alpha \Rightarrow \beta$ {β, α'dan tek bir türetim uygulanarak elde edilmiştir.}

$\alpha \overset{*}{\Rightarrow} \beta$ {β, α'dan 0 veya daha fazla kere türetim uygulanarak elde edilmiştir.}

Bir G CFG tarafından oluşturulan dil L(G) ile gösterilir.

$L(G) = \{w \mid w \in T^* \text{ ve } S \overset{*}{\underset{G}{\Rightarrow}} w\}$

CFL'de (Bağlamdan Bağımsız Dil) başlangıç sembolü ile başlanır. Herhangi bir türetim kuralı birçok kez kullanılabilir. Sonuçta elde edilen dizgi sadece sembollerden oluşur.

Örnek:

Bir G=({S},{a,b}, S, P) grameri
$S \rightarrow aSa$
$S \rightarrow bSb$
$S \rightarrow \lambda$
türetim kuralları ile bağlamdan bağımsızdır.
Bu gramere ait tipik bir türetim:

$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabbaa$
$L(G)=\{ww^R : w \in \{a,b\}^*\}$ (Palindrome) Bu dil düzgün değildir. Lineerdir fakat sağdan ya da soldan lineer değildir.

Örnek:

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

S=>abB=>abbbAa=>abbba

S=>abB=>abbbAa=>abbbaaBba
=>abbbaabbAaba=>abbbaabbaba

$L(G)= \{ab(bbaa)^n bba(ba)^n\}\ n \geq 0$

Bu gramer de lineer bir gramerdir.

Örnek:

$S \rightarrow SS$

$S \rightarrow (S)$

$S \rightarrow \lambda$

Bu gramer ile ( ) nasıl oluşturulur?

S=>(S)=>( )

(( ) ( )) nasıl oluşturulur?

S=>(S) 2
=>(SS) 1
=>((S) S) 2
=>((S)(S)) 2
=>(( )(S)) 3
=>(( ) ( )) 3

Sonuç olarak tüm düzgün diller bağlamdan bağımsızdır. Çünkü bağlamdan bağımsız gramerlerde, her kuralın sol tarafında tek bir değişken vardır. Tüm düzgün diller de, her dilbilgisi kuralının sol tarafında tek bir değişken olan gramerler ile oluşturulur.

Fakat Palindrome örneğinde olduğu gibi tüm Bağlamdan Bağımsız Gramerler düzgün değildir. Bu nedenle, düzgün diller bağlamdan bağımsız dillerin bir alt kümesidir.

Örnek:

$L=\{a^n b^m : n \neq m\}$ dilinin bağlamdan bağımsız olduğunu göstermek için bu dil için BB bir gramer (CFG) bulmalıyız.

İlk olarak n>m durumunu düşünelim. Eşit sayıda a ve b türetilir ve sol tarafa ekstradan a'lar eklenir.

$S \rightarrow AS_1$

$S_1 \rightarrow aS_1b \mid \lambda$

$A \rightarrow aA \mid a$

Aynı şey n<m için de düşünülür:

$S \rightarrow S_1B$

$S_1 \rightarrow aS_1b \mid \lambda$

$B \rightarrow bB \mid b$

Sonuç olarak:

$S \rightarrow AS_1 \mid S_1B$

$S_1 \rightarrow aS_1b \mid \lambda$

$A \rightarrow aA \mid a$

$B \rightarrow bB \mid b$

Bu gramer CFG'dir fakat lineer değildir.

Lineer olmayan CFG'lerde, bir türetim birden fazla değişken ile farklı şekillerde elde edilebilir. Bu gibi durumlarda, hangi değişkeni hangi sırada kullanacağımızı kendimiz seçebiliriz. Örneğin;

G=({A,B,S},{a,b},S,P) grameri için türetim kuralları:

1. S→AB
2. A→aaA
3. A→ λ
4. B→Bb
5. B→ λ   olsun.

Bu gramer $L(G)=\{a^{2n}b^m: n \geq 0, m \geq 0\}$ dilini oluşturur.

aab dizgisi için iki türetim şu şekilde olabilir:

S=>AB=>aaAB=>aaB=>aaBb=>aab

S=>AB=>ABb=>aaABb=>aaAb=>aab

Bunlar arasındaki tek fark kuralların farklı sırada uygulanmasıdır. Her ikisi de aynı kuralları kullanarak aynı cümleyi elde etmiştir. Bundan doğacak karışıklığı önlemek için değişkenleri belirli bir sırada yerleştiririz.

Tanım: Eğer her adımda en soldaki değişkeni yerleştirerek cümleyi elde ediyorsak buna «soldan türetme» (leftmost derivation) denir. Her adımda en sağdaki değişken yerleştiriliyorsa «sağdan türetme» (rightmost derivation) denir.

Örnek: $S \rightarrow aAB$

$A \rightarrow bBb$

$B \rightarrow A \mid \lambda$

türetim kurallarıyla verilmiş gramer için abbbb dizgisi;

Soldan türetme:

S=>aAB=>abBbB=>abAbB=>abbBbbB=>abbbbB=>abbbb

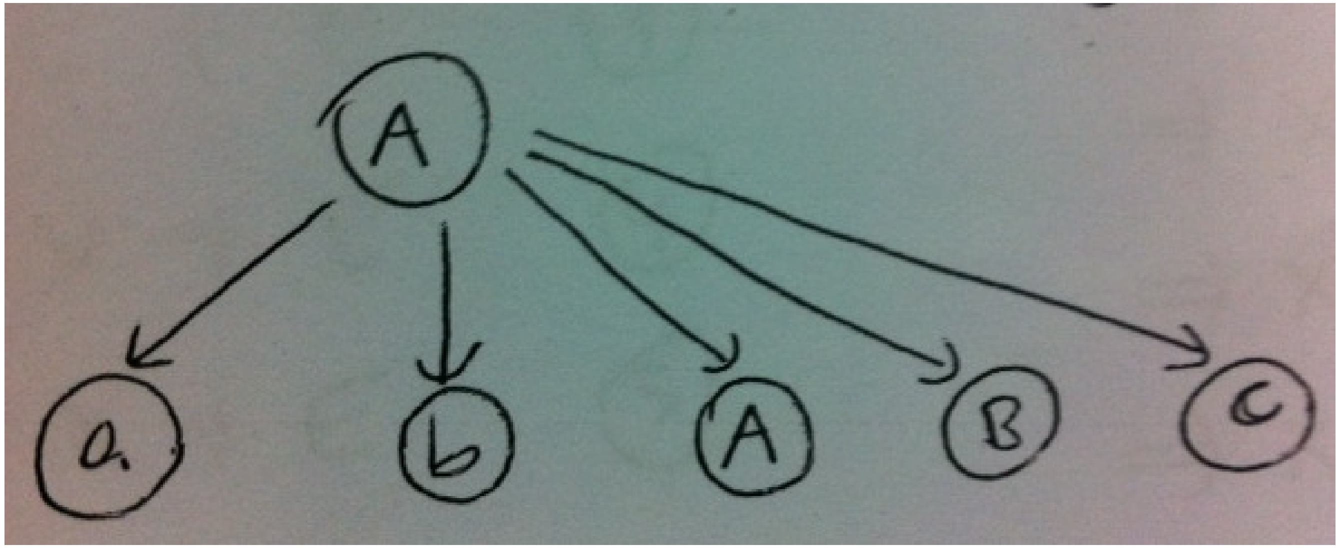
Sağdan türetme:

S=>aAB=>aA=>abBb=>abAb=>abbBbb=>abbbb

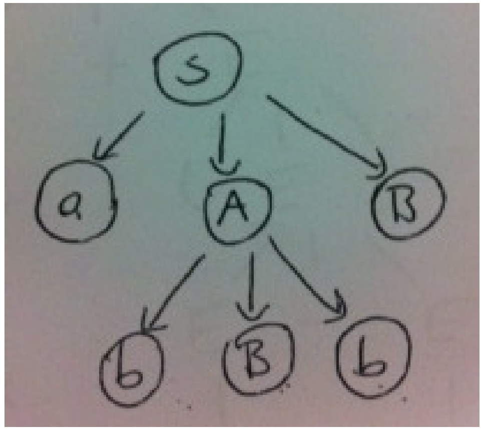
Türetim Ağaçları (Ayrıştırma Ağaçları)
Parse Trees (Derivation Trees)

Türetim kurallarının kullanılma sırasından bağımsız bir şekilde türetimleri göstermenin diğer bir yolu da türetim ağaçları kullanmaktır.

Örneğin; A→ abABc kuralını gösteren türetim ağacı:

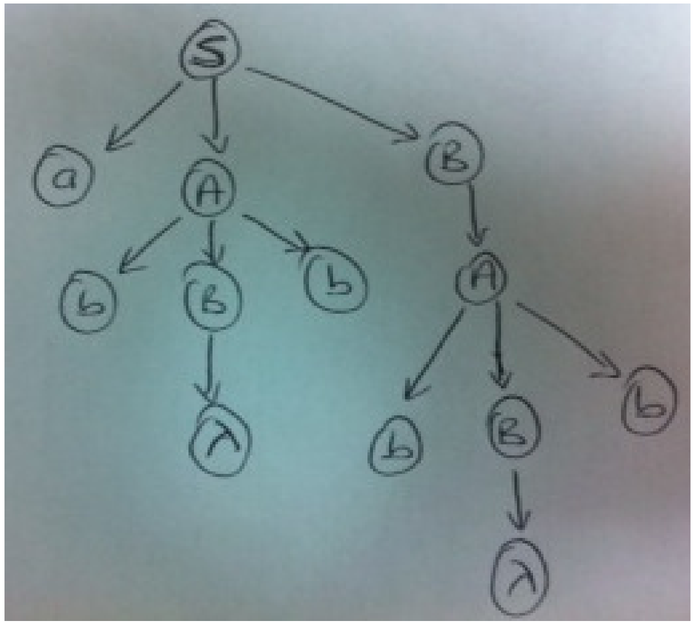


- Örneğin; S → aAB
  A → bBb
  B → A | λ türetim kurallarıyla verilen G grameri için;



Kısmi türetim ağacı
abBbB

Türetim Ağacı:



abbbb
dizgisi için

Örnek:

$E \rightarrow E+E$
$E \rightarrow E-E$
$E \rightarrow (E)$
$E \rightarrow V$
$V \rightarrow x$
$V \rightarrow y$
$V \rightarrow z$

Gramer kurallarını kullanarak x+(y-z) elde ediniz.

Soldan Türetme:

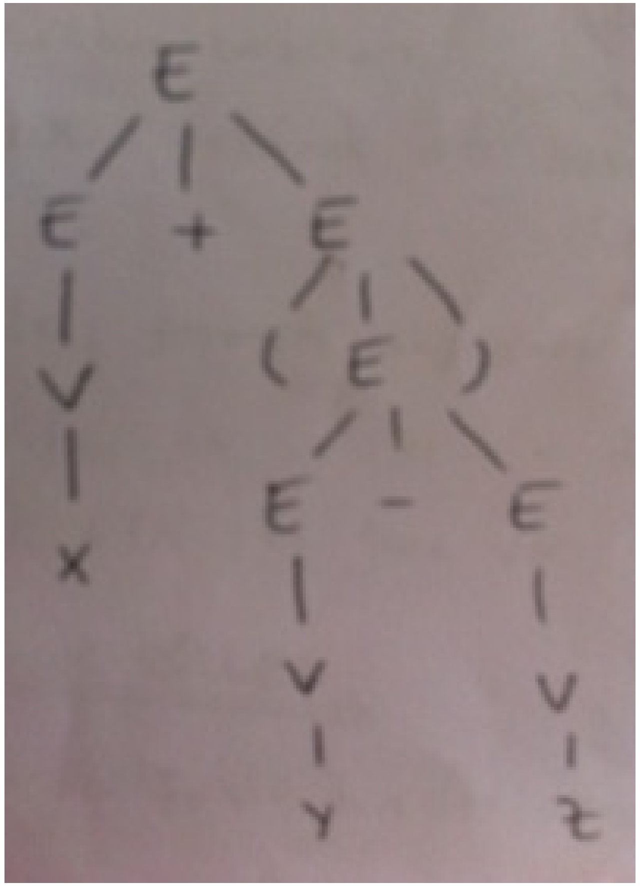
$$E \Rightarrow E+E \Rightarrow V+E \Rightarrow x+E \Rightarrow x+(E) \Rightarrow x+(E-E)$$
$$\Rightarrow x+(V-E) \Rightarrow x+(y-E) \Rightarrow x+(y-V) \Rightarrow x+(y-z)$$

Soldan Türetme:

$E \Rightarrow E+E \Rightarrow V+E \Rightarrow x+E \Rightarrow x+(E) \Rightarrow x+(E-E)$

$\Rightarrow x+(V-E) \Rightarrow x+(y-E) \Rightarrow x+(y-V) \Rightarrow x+(y-z)$

Türetim Ağacı:



- Sağdan Türetme:

$E \Rightarrow E+E \Rightarrow E+(E) \Rightarrow E+(E-E) \Rightarrow E+(E-V) \Rightarrow E+(E-z)$
$\Rightarrow E+(V-z) \Rightarrow E+(y-z) \Rightarrow V+(y-z) \Rightarrow x+(y-z)$

**Her türetim ağacı tek bir soldan türetime ve tek bir sağdan türetime sahiptir.

Tanım: Bağlamdan bağımsız bir G=(V,T,S,P) grameri A $\in$ V, a $\in$ T, X $\in$ V* ve (A,a) çifti P'de en fazla bir kere olmak şartıyla tüm türetim kuralları A→aX şeklinde ise basit gramerdir(s-grammar).

Örneğin, S → aS | bSS | c grameri s-gramerdir.

S → aS | bSS | aSS | c grameri s-gramer değildir.

Çünkü (S, a) çifti iki tane vardır.

Eğer G grameri bir s-gramer ise L(G) diline ait herhangi bir w dizgisi |w| adımda çözülebilir.

Örneğin, S→aS→bSS→c türetim kuralları ile verilen gramer, s-gramer olduğu için, tüm kurallar A→aX şeklindedir.

w=abcc dizgisini ele alalım. s-gramerin özelliğine göre (A,a) çifti P'de en fazla bir kere olma şartı ile, abcc dizgisindeki a'nın hangi kural ile türetilmesi gerektiğini biliriz. S→aS

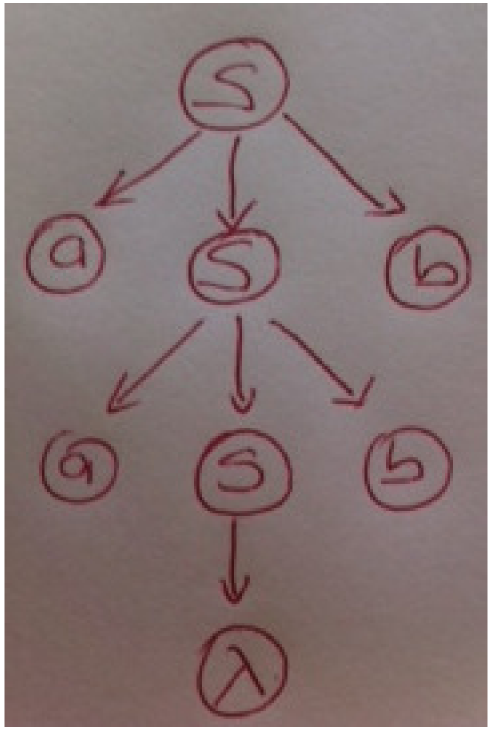
Benzer şekilde, b ve cc'yi türetmenin sadece tek bir yolu vardır. Bu nedenle w dizgisini |w| adımdan daha fazla adımda elde edemeyiz.

BELİRSİZ GRAMERLER VE DİLLER

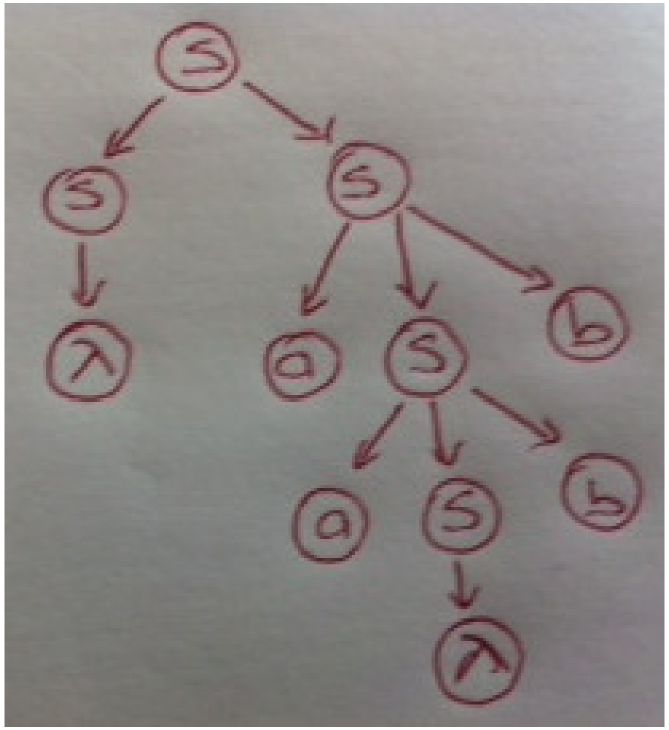
- Tanım: Bağlamdan bağımsız bir G grameri için $w \in L(G)$ olmak üzere en azından iki farklı türetim ağacı varsa (dizginin birden fazla soldan türetim ya da sağdan türetime sahip olması) bu gramer belirsizdir (ambiguity).

Örnek: S → aSb | SS | λ türetim kurallarıyla verilen gramer belirsizdir. aabb için iki türetim ağacı vardır:

$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$



$S \Rightarrow SS \Rightarrow S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$



- Örnek: G=(V, T, E, P) grameri V = {E, I}

T = {a, b, c, +, *, (,) }

Türetim kuralları: P:

E → I

E → E+E

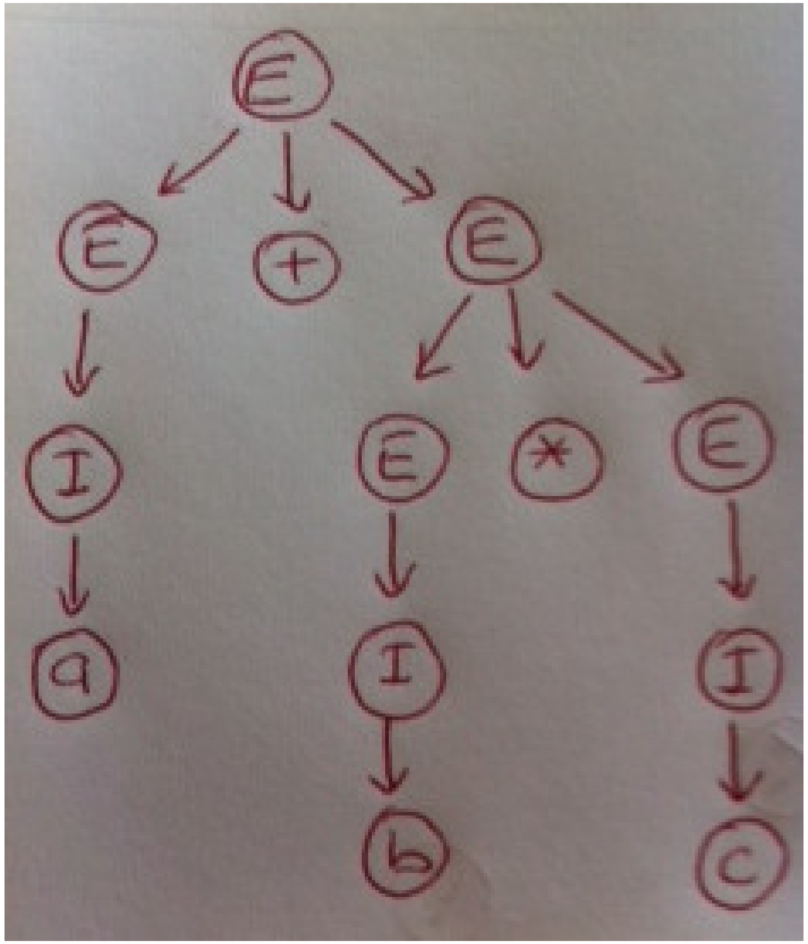
E → E*E

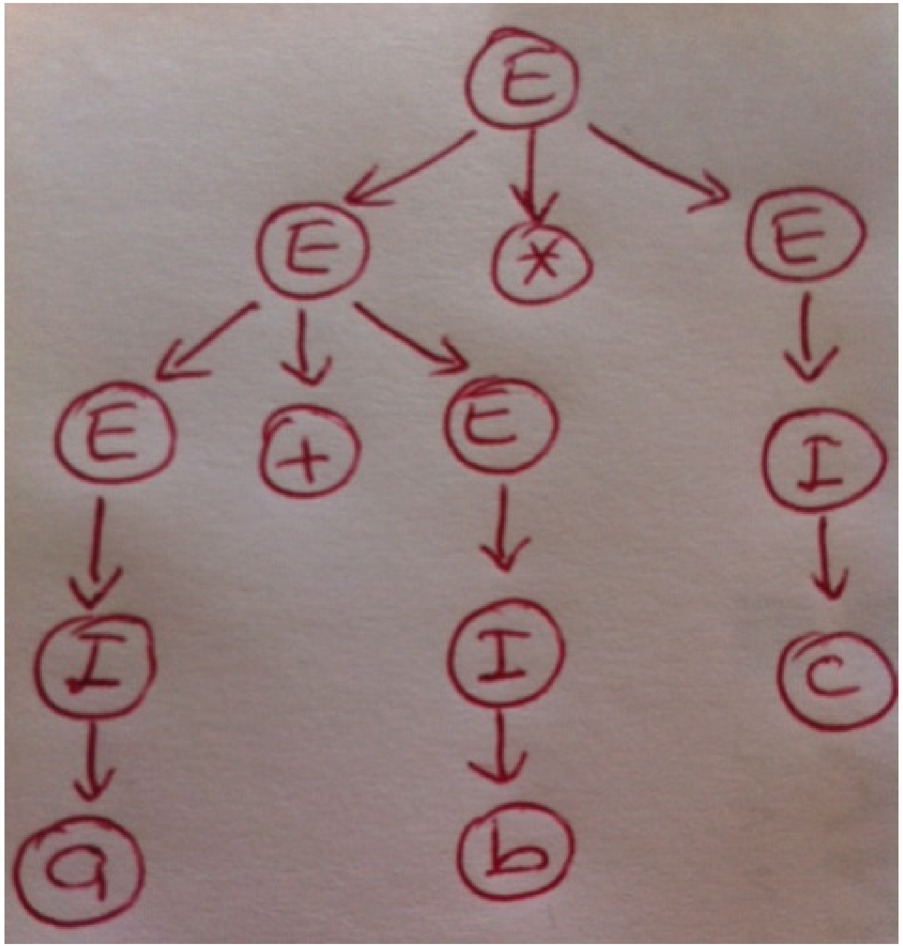
E → (E)

I → a|b|c

a+b*c dizgisi L(G) diline aittir ve iki türetim ağacı vardır. Bu nedenle G grameri belirsizdir:

a+b*c için iki farklı türetim ağacı:





- Örnek: V= {E, T, F, I} ve türetim kuralları:

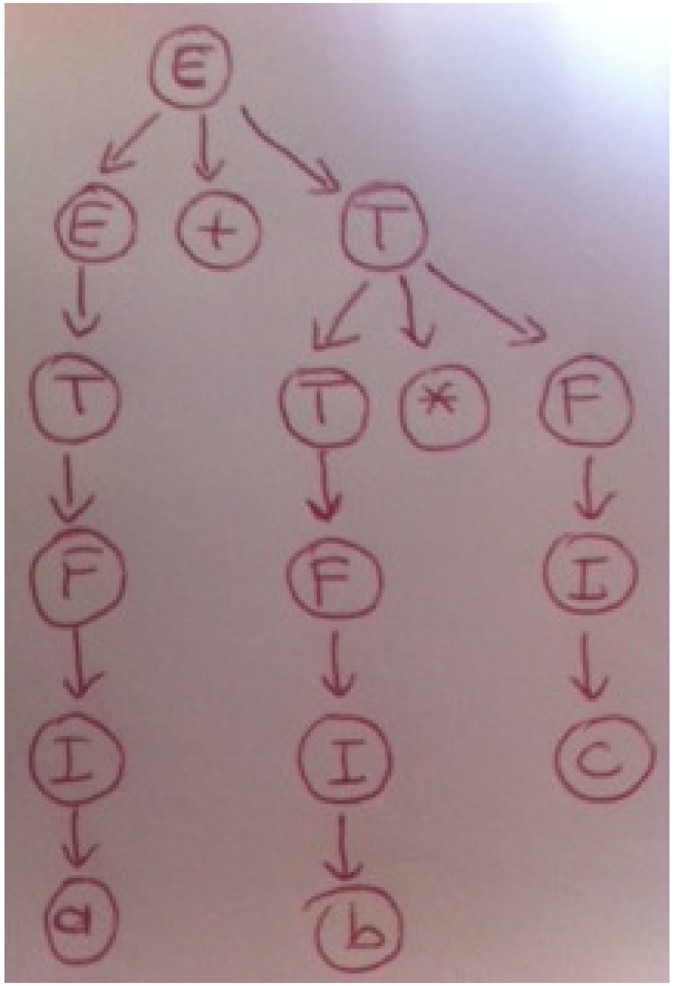
P: E → T | E+T

T → F | T*F

F → I | (E)

I → a|b|c

a+b*c için bu gramer belirsiz değildir (unambigious).



Gramerden kaynaklanan belirsizlik, buna eşdeğer bir gramer bulunarak ortadan kaldırılabilir. Ancak (bazı durumlarda) belirsizliğin dilde olması durumunda bu mümkün değildir.

Tanım: Bir L bağlamdan bağımsız dili için belirsiz olmayan bir gramer var ise, bu durumda L dili de belirsiz değildir! Eğer L dilini türeten her gramer belirsiz ise, bu dile inherently ambigious (niteliği gereği belirsiz) denir.

Örnek: $L = \{a^n b^n c^m\} \cup \{a^n b^m c^m\}$, $n,m \geq 0$ ve niteliği gereği belirsiz bağlamdan bağımsız dildir.

$L = L_1 \cup L_2$

L_1 şu şekilde oluşturulur: $S_1 \rightarrow S_1c \mid A$

$A \rightarrow aAb \mid \lambda$

L_2 : $S_2 \rightarrow aS_2 \mid B$

$B \rightarrow bBc \mid \lambda$

Sonuç olarak L dili $L: S \rightarrow S_1 \mid S_2$

Bu gramer belirsizdir çünkü $a^n b^n c^n$ dizgisi iki farklı türetime sahiptir. Bir tanesi $S \Rightarrow S_1$, diğeri ise $S \Rightarrow S_2$ ile başlar.

Örnek: S → AB | aaB

A → a | Aa

B → b

Gramerinin belirsiz olduğunu gösteriniz.

w=aab dizgisi için iki tane soldan türetme vardır:

$S \Rightarrow aaB \Rightarrow aab$

$S \Rightarrow AB \Rightarrow AaB \Rightarrow aaB \Rightarrow aab$

Örnek: $S \rightarrow aSb \mid SS \mid \lambda$ gramerinin belirsiz ancak bu gramere ait dilin belirsiz olmadığını gösteriniz.

ab dizgisi için gramerin belirsizliği:

$S \Rightarrow aSb \Rightarrow ab$

$S \Rightarrow SS \Rightarrow aSbS \Rightarrow abS \Rightarrow ab$

Buna eşdeğer belirsiz olmayan gramer:

$S \rightarrow A \mid \lambda$

$A \rightarrow aAb \mid ab \mid AA$

w = aabb dizgisi için $S \Rightarrow A \Rightarrow aAb \Rightarrow aaAbb \Rightarrow aabb$

w = abab dizgisi için $S \Rightarrow A \Rightarrow AA \Rightarrow abA \Rightarrow abab$

Bu dizgiler sadece tek bir şekilde türetilebilirler

CFG'lerin Basitleştirilmesi

Teorem: $G = (V,T,S,P)$ grameri bir CFG olsun. A ve B birbirinden farklı değişkenler olmak üzere, türetim kuralları P aşağıdaki şekilde verilsin:

$A \rightarrow x_1 B\, x_2$

$B \rightarrow y_1 \mid y_2 \mid \ldots \mid y_n$

$\hat{G}$ gramerinde $\hat{P}$ türetim kuralları, P'den $A \rightarrow x_1 B x_2$ kuralını çıkarıp $A \rightarrow x_1\, y_1\, x_2 \mid x_1\, y_2\, x_2 \mid \ldots \mid x_1\, y_n\, x_2$ kuralı eklenerek elde edilmiş ise $L(\hat{G}) = L(G)$. Bu iki gramer birbirinin eşdeğeridir.

Buna «Yerine Koyma Kuralı» (Substitution Rule) denir.

Örneğin, G = ({A,B}, {a,b,c}, A, P) için türetim kuralları:

P: A → a | aaA | abBc

B → abbA | b

B değişkenini yerine koyarak $\hat{G}$ grameri için türetim kuralları:

$\hat{P}$: A → a | aaA | ababbAc | abbc

B → abbA | b

Bu örnekte G ile $\hat{G}$ gramerleri eşdeğer gramerlerdir. aaabbc dizgisinin her iki gramerle elde edilmesi:

G gramerinde: $A \Rightarrow \mathrm{aaA} \Rightarrow \mathrm{aaabBc} \Rightarrow \mathrm{aaabbc}$

$\hat{G}$ $\mathrm{gramerinde}: A \Rightarrow \mathrm{aaA} \Rightarrow \mathrm{aaabbc}$

$\hat{G}$ gramerinde, B değişkeni ve ona ait türetim kuralları türetimde kullanılmadığı halde yer almaktadır. Bu şekilde kullanılmayan değişkenler ve türetim kuralları gramerden çıkartılabilir.

Kullanılmayan Değişkenlerin Çıkarılması

Tanım: Bir G=(V,T,S,P) CFG'de bir A $\in$ V değişkeninin kullanılır olabilmesi için x,y $\in$ (V ∪ T)* olmak üzere en azından bir tane $S \overset{*}{\Rightarrow} xAy \overset{*}{\Rightarrow} w$ olan w $\in$ L(G) dizgisi olmalıdır.

Örneğin, S → aSb | λ | A

A → aA

Türetim kuralları ile verilen gramerde S → A kuralı hiçbir rol oynamadığı için kaldırılabilir. Çünkü bir dizgi türetilemez. Bu durum gramere ait dili etkilemez ve basitleştirilmesini sağlar.

Örneğin, $S \rightarrow A$

$A \rightarrow aA \mid \lambda$

$B \rightarrow bA$

B değişkeni bu türetim kuralları ile verilen gramerde kullanılmaz. Bu nedenle $B \rightarrow bA$ kuralı gramerden çıkartılabilir.

$B \Rightarrow bA \Rightarrow baA \Rightarrow ba$ B'den dizgi üretilebildiği halde

$S \overset{*}{\Rightarrow} xBy \overset{*}{\Rightarrow}$ w hiçbir şekilde elde edilemez.

Sonuç olarak, bir X değişkeni iki şekilde gramerden çıkarılabilir:

- X bir dizgi üretemez
- X, başlangıç sembolü S'den erişilemeyen bir değişkendir.

Örnek: V={S,A,B,C} ve T={a,b} olmak üzere, G=(V,T,S,P) gramerinde kullanılmayan sembol ve kuralları eleyin.

P: S → aS | A | C

A → a

B → aa

C → aCb

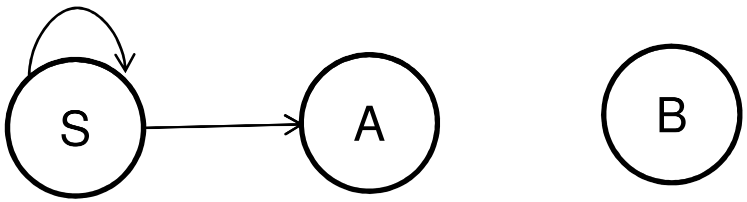
S ⇒ C ⇒ aCb ⇒ aaCbb ⇒ …

C değişkeni bir dizgi türetemediği için gramerden çıkartılabilir.

S → aS | A

A → a

B → aa



S'den B'ye erişebilmek mümkün değil

$\hat{G} = (\hat{V}, \hat{T}, S, \hat{P})$, $\hat{V} = \{S, A\}$, $\hat{T} = \{a\}$ olmak üzere,

$S \rightarrow aS \mid A$

$A \rightarrow a$

yerine koyma kuralı da uygulanarak:

$S \rightarrow aS \mid a$

elde edilir.

λ - Türetimlerinin Kaldırılması

Bir CFG'de $A \rightarrow \lambda$ şeklindeki kurala 'λ–türetimi' denir.

$A \overset{*}{\Rightarrow} \lambda$ ise A'ya 'geçersiz değişken' denir.

$A \rightarrow x_1x_2x_3x_4$ gibi bir türetim kuralında x_1 ve x_3 geçersiz değişkenler ise,

$A \rightarrow x_1x_2x_3x_4 \mid x_2x_3x_4 \mid x_1x_2x_4 \mid x_2x_4$ yazılabilir.

$$\left.\begin{array}{l} \text{Örneğin,}\quad S \rightarrow ABCD \\ \qquad\qquad A \rightarrow a \\ \qquad\qquad B \rightarrow \lambda \\ \qquad\qquad C \rightarrow ED \mid \lambda \\ \qquad\qquad D \rightarrow BC \\ \qquad\qquad E \rightarrow b \end{array}\right\}$$

Bu gramerde B, C ve D geçersiz değişkenlerdir.

O halde türetim kuralları şu şekilde yazılabilir:

$S \rightarrow ABCD \mid ABC \mid ABD \mid ACD \mid AB \mid AC \mid AD \mid A$

$A \rightarrow a$

$C \rightarrow ED \mid E$

$D \rightarrow BC \mid B \mid C$

$E \rightarrow b$

Örneğin, $S \rightarrow aS_1b$

$S_1 \rightarrow aS_1b \mid \lambda$

gramerinde, λ-türetimi kaldırılarak,

$S \rightarrow aS_1b \mid ab$

$S_1 \rightarrow aS_1b \mid ab$

yazılabilir.

Örnek: S → ABaC

A → BC

B → b |λ

C → D | λ

D → d

Türetim kuralları ile verilen gramere eşdeğer λ-türetimi olmayan CFG bulun.

Geçersiz değişkenler : A, B, C'dir.

S → ABaC | BaC | AaC | ABa | aC | Aa |Ba | a

A → BC | B | C

B → b

C → D

D → d

Birim Türetimlerin Kaldırılması

A ve B değişkenler olmak üzere, A→B şeklindeki türetim kuralına 'birim türetim' denir. (unit-production)

Örneğin;

$S \rightarrow Aa \mid B$

$B \rightarrow A \mid bb$

$A \rightarrow a \mid bc \mid B$

Birim türetimler:

$S \overset{*}{\Rightarrow} A$

$S \overset{*}{\Rightarrow} B$

$B \overset{*}{\Rightarrow} A$

$A \overset{*}{\Rightarrow} B$

Birim türetimler çıkarılır:

$S \rightarrow Aa$

$B \rightarrow bb$

$A \rightarrow a \mid bc$

Birim türetimler için ($S \overset{*}{\Rightarrow} A$, $S \overset{*}{\Rightarrow} B$, $B \overset{*}{\Rightarrow} A$, $A \overset{*}{\Rightarrow} B$)

Yeni kurallar belirlenir:

$S \rightarrow a \mid bc \mid bb$

$B \rightarrow a \mid bc$

$A \rightarrow bb$

Eşdeğer gramer:

$S \rightarrow Aa \mid a \mid bc \mid bb$

$B \rightarrow a \mid bc \mid bb$

$A \rightarrow a \mid bc \mid bb$

elde edilir.

Bu şekilde B değişkeni ve buna ait kurallar kullanım dışı olmuştur.

**Sonuç olarak CFG'lerde 3 şekilde basitleştirme yapılabilir, sırasıyla:

→ λ-türetimler

→ Birim türetimler

→ Kullanılmayan değişkenlerin gramerden çıkarılması

Örnek:

S→ aA | aBB

A→ aaA | λ

B→ bC | bbC

C→ B

Sadeleştirin ve bu gramere ait dili bulun.

λ-türetimlerin kaldırılması:

S→ aA | a | aBB

A→ aaA | aa

B→ bC | bbC

C→ B

Birim türetimlerin kaldırılması:

S→ aA | a | aBB

A→ aaA | aa

B→ bC | bbC

C→ bC | bbC

Sonuç olarak B ve C kullanılmayan değişkenlerdir.

S→ aA | a

A→ aaA | aa

Dili : L((aa)*a)

Örnek:

S→ A | B | C

A→ aAa | B

B→ bB | bb

C→ aCaa | D

D→ baD | abD |λ

λ-türetimleri ve birim türetimleri kaldırın.

Geçersiz değişkenler: D, C, S (D→ λ, C→D→ λ, S→C→ λ)

S→ A | B | C | λ

A→ aAa | B

B→ bB | bb

C→ aCaa | aaa | D

D→ baD | ba | abD |ab

**Gramerin başlangıç değişkeni (S) yok edilebilir bir değişken olduğuna göre gramerin türettiği dil λ'yı içermektedir. Bu nedenle S→λ olmalı.

Birim türetimlerin kaldırılması:

$C \rightarrow D$, $A \rightarrow B$, $S \rightarrow A$, $S \rightarrow B$, $S \rightarrow C$, $S \rightarrow C \rightarrow D$

(C,D) (A,B) (S,A) $S \rightarrow A \rightarrow B$(S,B) (S,C) (S,D)

$S \rightarrow$ aAa | bB | bb | aCaa| aaa | baD | ba |abD | ab|λ

$A \rightarrow$ aAa | bB | bb

$B \rightarrow$ bB | bb

$C \rightarrow$ aCaa | aaa | baD | ba | abD | ab

$D \rightarrow$ baD | ba | abD | ab

Örnek: Aşağıdaki grameri basitleştirin:

S→ aAa | bBb | λ

A→ C | a

B→ C | b

C→ CDE | λ

D→ A | B | ab

λ-türetimleri kaldıralım: (C, A, B, D) geçersiz değişkenler

S→ aAa | aa | bBb | bb | λ

A→ C | a

B→ C | b

C→ CDE | DE | CE | E

D→ A | B | ab

E için herhangi bir kural tanımlanmadığı için çıkarılabilir. Bu durumda C için de bir kural tanımlanmayacağı için bu değişken de çıkarılır.

S→ aAa | aa | bBb | bb | λ

A→ a

B→ b

D→ A | B | ab

Birim türetimleri çıkaralım: D→A, D→B

S→ aAa | aa | bBb | bb | λ

A→ a

B→ b

D→ a | b | ab

D değişkeni kullanılmayan değişkendir. S'den başlayıp D kullanılarak dizgi türetilemez. D'yi çıkaralım.

$S \rightarrow aAa \mid aa \mid bBb \mid bb \mid \lambda$

$A \rightarrow a$

$B \rightarrow b$

elde edilir.

Örnek: Aşağıdaki gramerde birim türetimler kaldırılınca elde edilen gramer nasıldır?

S→ A | B

A→ B | C | aB | b

B→ C

C→ B | Aa

S→ Aa | aB | b

A→ Aa | aB | b

B→ Aa

C→ Aa

C de çıkarılabilir.

Örnek:

S→ ABC

A→ aA | B

B→ λ | C

C→ c | cC

λ-türetimlerin kaldırılması : Geçersiz değişkenler: B, A

S→ ABC | AC | BC | C

A→ aA |a |B

B→ C

C→ c | cC

Birim türetimlerin kaldırılması: $S \rightarrow C$, $A \rightarrow B$, $B \rightarrow C$

$S \rightarrow ABC \mid AC \mid BC \mid c \mid cC$

$A \rightarrow aA \mid a \mid c \mid cC$

$B \rightarrow c \mid cC$

$C \rightarrow c \mid cC$

elde edilir.