BLG311 Biçimsel Diller ve Otomatlar

Biçimsel Dillerin Matematiksel Temelleri

A.Emre Harmancı Tolga Ovatman Berk Canberk

2017

İçerik

Biçimsel Diller

Dil

Dil, insanların iletişim amaçlısistemler kullanma ve türetme yeteneğinin somutlaşmış hali ya da bu bağlamda oluşturulan bir iletişim sistemine verilen isimdir.

Biçimsel Diller

Biçimsel Dil

Matematik, bilgisayar bilimleri ve dilbiliminde biçimsel dil, sembol katarlarından oluşan bir küme olarak tanımlanmıştır. Bir biçimsel dilin alfabesi, o dilde sözcükler olarak kullanılan katarlarıoluşturan sembolleri, harfleri ya da genel olarak tekilleri(token) içerir.

Tümevarımla Tanımlanan Kümeler

Biçimsel dil oluşturmada kullanılan yöntemlerden biri tümevarım kullanarak bir semboller kümesinden bir sözcükler kümesi tanımlamaktır.

Tümevarımla Tanımlama

Bir terimin tanımlanmasında terimin kendisinin sonsuz türetim ortaya çıkarmayacak biçimde kullanılmasıdır. özyineli tanım da denir.

Tümevarımla Tanımlanan Kümeler

Bir E kümesini tümevarımla tanımlamak için:

- Taban: Yapıtaşı elemanları kümesi veya taban S ile gösterilsin. Bu kümenin içindeki elemanlardan hareketle diğer elemanlar türetilebiliyorsa, bu elemanlara yapıtaşı elemanları denir. Örnek olarak alfabe bir dil kümesinin yapıtaşı elemanları kümesidir.
- Kurallar: Bir E kümensinin içindeki elemanlardan hareketle yeni elemanları türetmenin kurallarıdır. Bir dil, alfabeden belli kurallar neticesinde türetilmiş sözcükler topluluğudur (kümesidir).

$$\begin{aligned} \Omega = \{f_1, f_2, \ldots, f_n\} \wedge \forall f_i \in \Omega \\ f_i : E \times E \times \ldots \times E \rightarrow E \\ \forall x_1, x_2, \ldots, x_p \in E \quad f_i(x_1, x_2, \ldots, x_p) = X \in E \end{aligned}$$

- Kapanış: E kümesi, taban elemanları ve sadece kurallar uygulanması sonucu oluşan elemanlardan başka bir elemanı olmayan kümedir. $\Omega(E) \subseteq E$

Tümevarımla Tanımlanan Kümeler

Bir E kümesini tümevarımla tanımlamak için:

- Tümevarım:

$$
\begin{aligned}
&S_0 = S \\
&S_1 = \Omega(S_0) \cup S_0 \text{ kurallar uygulanır} \\
&S_2 = \Omega(S_1) \cup S_1 \\
&\dots \\
&S_{i+1} = S_i \cup \Omega(S_i) = S_i \subseteq E \text{ (kapanış)}
\end{aligned}
$$

  i sonlu ya da sonsuz olabilir.

- Elemanın yüksekliği: Elemanın ilk defa kaçıncı sırada elde edildiğini gösterir. Sözcüğün uzunluğudur. $H(x) = min\{i|x \in S_i\}$

Tümevarımla Tanımlanan Kümeler

çift sayılar kümesini tümevarımla oluşturalım :

i $0 \in E$ taban

ii $n \in E \Rightarrow (n+2) \in E$

iii Bu kuralın uygulanması ile elde edilenler dışında hiç bir sayı çift sayı değildir.

Tanımlar

- Alfabe: Sonlu, boş olmayan simgeler veya karakterler kümesi Σ
- Sözcük: Σ'nın sonlu bir dizisi veya katarı.
- Sözcük uzunluğu: Sözcükteki simge sayısı.
- Boş sözcük: Uzunluğu sıfır olan sözcük. Λ or ε.

Sözcük Bitiştirme

Sözcük bitiştirme iki karakter katarının art arda eklenmesiyle oluşturulur.

$$(x = a_1 a_2 \ldots a_n) \wedge (y = b_1 b_2 \ldots b_m) \Rightarrow xy = a_1 a_2 \ldots a_n b_1 b_2 \ldots b_m (x\&y)$$

Bitiştirmede birim eleman Λ'dır. $x = \Lambda \Rightarrow xy = y$
$y = \Lambda \Rightarrow xy = x$

Boş katar ve birleşme özelliği (asosiyativite) gösteren bitiştirme işlemi (konkatenasyon) ile alfabe üzerinde oluşturulmuş sözcükler kümesi (Σ^*) bir monoiddir. Bundan sonra aşağıda verilmiş olan kümeler bilgisayar biliminde sıklıkla kullanılır.

Katar tersi

$(baba)^R = abab$

$(ana)^R = ana$

Ters katar tanımı (tümevarım ile):

1. $|w| = 0 \Rightarrow w^R = w = \Lambda$
2. w ve u katarlarımız olsun.

   $|u| = n$ $\qquad |w| = n+1$

   $n \in \mathbb{N}$. Bu varsayımlarla

   $(w = ua) \wedge (a \in \Sigma) \Rightarrow w^R = au^R$

örneğin:

$$|w| = 1 \Rightarrow w = \Lambda a$$
$$w^R = a\Lambda = w$$
$$|w| = 2 \Rightarrow w = ua(u = b)$$
$$w^R = au^R = au = ab$$
$$|w| = 3 \Rightarrow w = ua, u = cb$$
$$w^R = au^R = abc$$

Katar tersi

Teorem

$(wx)^R = x^R \cdot w^R \wedge |x| = n, |w| = m \wedge m, n \in \mathbb{N}$

İspat.

Taban adım:

$|x| = 0 \Rightarrow x = \Lambda$

$(wx)^R = (w\Lambda)^R = w^R = \Lambda w^R = \Lambda^R w^R = x^R w^R$

Tümevarım adımı: $|x| \leq n \Rightarrow (wx)^R = x^R w^R$

$|x| = n+1$ için

$(x = ua) \wedge (|u| = n) \wedge (a \in \Sigma) \wedge (x^r = au^r)$

$(wx)^R = (w(ua))^R = ((wu)a)^R = a(u^R w^R) = au^R w^R = x^R w^R$

$\square$

örneğin (snow ball)^R= (ball)^R(snow)^R

Σ^+, Σ alfabesi üzerinde tanımlı boş olmayan sözcükleri temsil etsin

i $a \in \Sigma \Rightarrow a \in \Sigma^+$

ii $(x \in \Sigma^+ \wedge a \in \Sigma) \Rightarrow ax \in \Sigma^+$

iii i ve ii'nin sonlu uygulamaları dışında elde edilemeyen hiç bir eleman Σ^+'nın elemanı olamaz

örneğin:
$\Sigma = \{a, b\} \Rightarrow \Sigma^+ = \{a, b, aa, ba, ab, bb, aaa, aab, \ldots\}$.

Σ^*, Σ alfabesi üzerinde tanımlı tüm sözcükleri temsil etsin

i $\Lambda \in \Sigma^*$

ii $(x \in \Sigma^* \wedge a \in \Sigma) \Rightarrow ax \in \Sigma^*$

iii i ve ii'nin sonlu uygulamaları dışında elde edilemeyen hiç bir eleman Σ^*'nın elemanı olamaz

örneğin:
$\Sigma = \{a, b\} \Rightarrow \Sigma^* = \{\Lambda, a, b, aa, ba, ab, bb, aaa, aab, \ldots\}$
$\Sigma = \{0, 1\} \Rightarrow \Sigma^* = \{\Lambda, 0, 1, 00, 01, 10, 11, 000, 001, \ldots\}$

$(x \in \Sigma^*) \land (\forall n \in \mathbb{N})$ olarak kabul edildiğine, bir katarın n'inci kuvveti(x^n) aşağıdaki gibi tanımlanabilir:

1. $x^0 = \Lambda$
2. $x^{n+1} = x^n \cdot x = (x^n \& x)$

örneğin:
$\Sigma = \{a, b\} \;\; x = ab$
$x^0 = \Lambda, \;\; x^1 = ab, \;\; x^2 = abab, \;\; x^3 = ababab$

ya da
$\{a^n b^n | n \geq 0\}$ ile tanımlanan küme: $\{\Lambda, ab, aabb, aaabbb, \ldots\}$

Σ sonlu bir alfabe olsun.

Σ üzerinde tanımlanan her dil Σ^*'nin bir alt kümesidir.

Σ^*'den bir alt küme seçmenin kuralları bir gramer yardımı ile tanımlanabilir.

örneğin:
$\{a^m b^n | m, n \in \mathbb{N}\}$ $\{a, b\}$ kümesi üzerinde tanımlanmış olsun.

Bu dilde a ve b harici sembol bulunamaz.

Bu dilde b her zaman a'dan once gelir. örneğin ba bu dile ait değildir.

Çarpım diller

A ve B Σ üzerine tnaımlı diller olsun. $A \times B$ kartezyen çarpımı yeni bir dil tanımlar.
$AB = \{xy | x \in A \wedge y \in B\}$
$AB = \{z | z = xy \wedge x \in A \wedge y \in B\}$

örneğin:
$\Sigma = \{a, b\}$ alfabe olsun
$A = \{\Lambda, a, ab\}$ ve $B = \{a, bb\}$ olarka tanımlandıysa
$AB = \{a, bb, aa, abb, aba, abbb\}$
$BA = \{a, aa, aab, bb, bba, bbab\}$
$AB \neq BA$

Teoremler

$\varnothing$ boş dili teems etsin; A,B,C,D Σ alfabesi üzerinde tanımlı çeşitli diller olsun.

1. $A\varnothing = \varnothing A = \varnothing$
2. $A\{\Lambda\} = \{\Lambda\}A = A$
3. $(AB)C = A(BC)$
4. $(A \subset B) \wedge (C \subset D) \Rightarrow AC \subset BD$
5. $A(B \cup C) = AB \cup AC$
6. $(B \cup C)A = BA \cup CA$
7. $A(B \cap C) \subset AB \cap AC$
8. $(B \cap C)A \subset BA \cap CA$

Seçilen İspatlar.

$$A\varnothing = \varnothing A = \varnothing$$

$A\varnothing = \{xy | x \in A \wedge y \in \varnothing\}$ mümkün değildir çünkü $\forall y; y \notin \varnothing$.

$A\varnothing = \varnothing$'yi sağlayan bir x, y çifti bulunmaz. $\square$

Seçilen İspatlar.

$$(A \subset B) \wedge (C \subset D) \Rightarrow AC \subset BD$$

Doğrudan ispat yöntemini kullanacağız. Kabulümüz
$A \subset B \wedge C \subset D$ and $z \in AC$

$$z = xy \Leftrightarrow x \in A \wedge y \in C$$
$$(A \subset B \wedge x \in A \Rightarrow x \in B) \wedge (C \subset D \wedge y \in C \Rightarrow y \in D)$$
$$(x \in B \wedge y \in D) \Leftrightarrow xy \in BD$$
$$(xy \in AC \Rightarrow xy \in BD) \Leftrightarrow AC \subset BD$$

$\square$

Seçilen İspatlar.

$$A(B \cap C) \subset AB \cap AC$$

$$\begin{aligned}
A(B \cap C) &\Rightarrow \forall z(z = xy \wedge x \in A \wedge y \in B \cap C) \\
x \in A \wedge y \in B \cap C &\Leftrightarrow x \in A \wedge y \in B \wedge y \in C \\
x \in A \wedge y \in B \wedge y \in C &\Leftrightarrow x \in A \wedge y \in B \wedge x \in A \wedge y \in C \\
x \in A \wedge y \in B \wedge x \in A \wedge y \in C &\Leftrightarrow xy \in AB \wedge xy \in AC \\
xy \in AB \wedge xy \in AC &\Rightarrow xy \in (AB \cap AC) \\
\forall z(z \in A(B \cap C) &\Rightarrow z \in AB \cap AC)
\end{aligned}$$

$\square$

$AB \cap AC \subseteq A(B \cap C)$'nın her zaman doğru olmadığını göstermek için bir örnek sunalım.

$A = \{a^n | n \in \mathbb{N}\}$
$B = \{a^n b^n | n \in \mathbb{N}\}$
$C = \{b^n | n \in \mathbb{N}\}$

$z = a^5 b^2$. $z \in AB \cap AC$ olsun.
$B \cap C = \{\Lambda\}$ olduğu için
$z \notin A(B \cap C)$.

$z = a^5 b^2 = a^3 a^2 b^2$
$z \in AB \cap AC$
$z \notin A(B \cap C)$ çünkü $B \cap C = \{\Lambda\}$

A^n : A dili Σ üzerinde tanımlansın

i $A^0 = \{\Lambda\}$

ii $A^{n+1} = A^n A; \forall n \in \mathbb{N}$

örneğin:

$\Sigma = \{a, b\}$

$A = \{\Lambda, a, b\}$

$A^1 = A$

$A^2 = \{\Lambda, a, b, aa, ab, ba, bb\}$

Teoremler

A ve B Σ üzerinde tanımlı farklı diller olsun.

1. $A^m A^n = A^{m+n}$
2. $(A^m)^n = A^{mn}$
3. $A \subset B \Rightarrow A^n \subset B^n \ldots$

İspat.

$$A^m A^n = A^{m+n}$$

Tümevarım kullanacağız.
Taban durum için önceki tanımımızı kullanalım: $n = 1 : A^m A^1 = A^{m+1}$

Tümevarım adımı: $A^m A^{n+1} = A^m A^n A^1$
Bitiştirme işlemi asosyatiftir: $A^m(A^n A^1) = (A^m A^n)A^1 = A^{m+n}A^1$ $\square$

İspat.

$$(A^m)^n = A^{mn}$$

Tümevarım kullanacağız.
Taban durum için $n = 1 : A^m = A^m$

Tümevarım adımı: $(A^m)^{n+1} = (A^m)^n(A^m)^1 = A^{mn+m} = A^{m(n+1)}$

$\square$

İspat.

$$A \subset B \Rightarrow A^n \subset B^n \ldots$$

Hatırlayalım: $A^2 = A \times A, B^2 = B \times B$ ve $A \subset B \Rightarrow (\forall x \in A \rightarrow x \in B)$

$n = 2$ için:
$A^2 = \{xy | x \in A \wedge y \in A\}$
$A \subset B \Rightarrow (x \in A \rightarrow x \in B)$
$A \subset B \Rightarrow (y \in A \rightarrow y \in B)$

$(\forall xy \in A^2 \rightarrow xy \in B^2) \Rightarrow A^2 \subset B^2$
İspat $n := n + 1$ için de benzer biçimde devam eder $\square$

Yıldız kapanışı(Kleene Yıldızı, Kleen Kapanışı)

Pozitif Kapanış

$A^+ : \bigcup_{n=1}^{\infty} A^n = A \cup \dots$

Yıldız Kapanışı

$A^* : \bigcup_{n=0}^{\infty} A^n = A^0 \cup A \cup \dots$

Teoremler

A ve B Σ üzerine tanımlı farklı diller olsun ve $n \in \mathbb{N}$ olsun.

1. $A^* = \Lambda \cup A^+$. Tanımdan türemiştir
2. $A^n \subseteq A^*, n \geq 0$. Tanımdan türemiştir
3. $A^n \subseteq A^+, n \geq 1$. Tanımdan türemiştir
4. $A \subseteq AB^*$ İpucu: $A^0 \subseteq B^* \Rightarrow A \subseteq AB^*$

Teoremler

A ve B Σ üzerine tanımlı farklı diller olsun ve $n \in \mathbb{N}$ olsun.

5. $A \subseteq B^*A$
6. $A \subseteq B \Rightarrow A^* \subseteq B^*$. $A \subseteq B \Rightarrow A^n \subseteq B^n$ kullanılarak ispatlanabilir. Tüm n'ler için doğruysa, tüm n'lerin birleşimi için de doğrudur.
7. $A \subseteq B \Rightarrow A^+ \subseteq B^+$
8. $AA^* = A^*A = A^+$
9. $\Lambda \in A \Leftrightarrow A^+ = A^*$
10. $(A^*)^* = A^*A^* = A^*$
11. $(A^*)^+ = (A^+)^* = A^*$
12. $A^*A^+ = A^+A^* = A^+$
13. $(A^*B^*)^* = (A \cup B)^* = (A^* \cup B^*)^*$.

Seçilen İspatlar.

$$AA^* = A^*A = A^+$$

$$AA^* = A^+ \Rightarrow A(A^0 \cup A^1 \cup \ldots) = A \cup A^2 \cup \ldots = A^+ \qquad \square$$

Seçilen İspatlar

$$\Lambda \in A \Leftrightarrow A^+ = A^*$$

önce $\Lambda \in A \Rightarrow A^+ = A^*$ olduğunu gösterelim

$\Lambda \in A \Rightarrow A \cup \{\Lambda\} = A$

$A \cup A^0 = A$

$A^+ = A \cup \{\bigcup_{n=2}^{\infty} A^n\} = A^0 \cup A \cup \{\ldots\} = A^*$

Seçilen İspatlar.

$$\Lambda \in A \Leftrightarrow A^+ = A^*$$

İkincil olarak $A^+ = A^* \Rightarrow \Lambda \in A$

Eğer $A \cup A^2 \cup \ldots = A^0 \cup A \cup \ldots$
$A^0 \subseteq A \cup A^2 \cup \ldots$

Bu içerme iki şekilde yorumlanabilir:

- i $\Lambda \in A \Rightarrow A^0 \subseteq A$
- ii $\Lambda \notin A \Rightarrow \exists i, \Lambda \in A^i, i \in (\mathbb{N}^+ - 1)$

Oysa $\Lambda \notin A \Rightarrow \forall x \in A^i(|x| \geq i)$
öte yandan $|\Lambda| = 0 \Rightarrow \Lambda \notin A^i \wedge i \geq 2$.

Bu çelişki ii'yi yanlış yapar. Bu halde i doğru olmalıdır.

$\square$

Seçilen İspatlar

$$(A^*B^*)^* = (A \cup B)^*$$

$(A^*B^*)^* \subseteq (A \cup B)^*$ ve $(A \cup B)^* \subseteq (A^*B^*)^*$ ispatlanmalı

$$A \subseteq A \cup B \wedge B \subseteq A \cup B$$
$$A^* \subseteq (A \cup B)^* \wedge B^* \subseteq (A \cup B)^*$$
$$A^*B^* \subseteq (A \cup B)^*$$
$$(A^*B^*)^* \subseteq ((A \cup B)^*)^*$$
$$(A^*B^*)^* \subseteq (A \cup B)^*$$

Seçilen İspatlar.

$$(A^*B^*)^* = (A \cup B)^*$$

$(A^*B^*)^* \subseteq (A \cup B)^*$ ve $(A \cup B)^* \subseteq (A^*B^*)^*$ ispatlanmalı

$A \subseteq A^* \subseteq A^*B^* \wedge B \subseteq B^* \subseteq A^*B^*$
A^* ve B^* Λ içerdiği için yukarıdaki ifadeler yazılabilir

$(A \cup B \subseteq A^*B^*) \Rightarrow (A \cup B)^* \subseteq (A^*B^*)^*$
$(A^*B^*)^* \subseteq (A \cup B)^*$ ve $(A \cup B)^* \subseteq (A^*B^*)^*$. $\square$

Arden Teoremi

A ve B Σ^*'nin alt kümeleri olsun ve $\Lambda \notin A$ olsun.
$X = AX \cup B$'nın tek çözümü $X = A^*B$

$$X = AX \cup B$$
$$X = A(AX \cup B) \cup B$$
$$X = A^2X \cup AB \cup B$$
$$X = A^2(AX \cup B) \cup AB \cup B$$
$$X = A^3X \cup A^2B \cup AB \cup B$$
$$\ldots$$
$$X = B \cup AB \cup A^2B \cup A^3B \ldots$$
$$X = B(A^0 \cup A \cup A^2 \cup A^3 \ldots)$$
$$X = B(A^0 \cup A \cup A^2 \cup A^3 \ldots)$$
$$X = BA^*$$

Örnek

$X = A^*B$ bir çözümdür
$\{\Lambda\}B = B$ olduğundan,

$$\begin{aligned} A^*B &= AA^*B \cup B \\ &= A^+B \cup B \\ &= (A^+ \cup \{\Lambda\})B \\ &= A^*B \end{aligned}$$

Biçimsel Gramer

Biçimsel gramerler bir biçimsel dildeki katarların oluşturulmasını tanımlayan kurallar bütünüdür. Bu kurallar, dilin alfabesinin kullanılarak dildeki katarların dilin sözdizimine uygun olarak nasıl oluşturulabileceğini tanımlarlar.

Örneğin:
Aritmetik işlemleri oluşturmak için aşağıdaki kurallar kullanılabilir:
$\mathbb{Z} \ + \ - \ \times \ / \ (\ )$. Sıfıra bölüm hariç bütün doğru tanımlanmış aritmetik ifadeler anlamlıdır.
$(((2-1)/3)+4\times 6)$ doğru oluşturulmuş ve anlamlı bir aritmetik ifadedir. $2+(3/(5-(10/2)))$ da doğru oluşturulmuştur fakat sıfıra bölüm nedeniyle anlamlı değildir.

Gramer sözdizimi

Biçimsel gramerlerin sınıflandırılması ilk defa 1950li yıllarda Noam Chomsky tarafından önerilmiştir. Buna göre bir cümle yapısı grameri (phrase structure grammar) G biçimsel olarak $(N, \Sigma, n_0, \mapsto)$ çoklusuyla ifade edilir:

- N *uç olmayan ara simgeler* kümesi.
- Σ bir *uç simgeler* kümesi. N ile ayrık bir kümedir.
- n_0 başlangıç simgesi.
- $\mapsto$ üretim kuralları.
  $V = N \cup \Sigma$, $\mapsto$ bağıntısı V^* üzerine tanımlanmış olsun.
  $V^*NV^* \to V^*$ yapısına sahiptir.
  Burada $*$ Kleene yıldızı işlecidir.

Gramerlerin semantik yapısı

Bir gramerin işleyişi katarlar üzerinde tanımlı bağıntılar ile temsil edilebilir:

- Doğrudan türetme bağıntısı $G = (N, \Sigma, n_0, \mapsto)$ gramerinde V^*, $\Rightarrow_G$ olarak gösterilir ve şöyle tanımlanır:
  $x \Rightarrow_G y \leftrightarrow \exists l, w, r \in V^* : (x = lwr) \wedge (w \rightarrow w' \in \mapsto) \wedge (y = lw'r)$
- $\Rightarrow_G$'un yansımalı geçişli kapanışı bize erişilebilirlik bağıntısını $\Rightarrow^*{}_G$ verir.
  Doğrudan türetimin erişilebilirliği sonlu adımlı bir şekilde türetilebilecek sözcükleri verir.
  Bu nedenle $\sigma \in \Sigma^*$ in $n_0 \Rightarrow^*{}_G \sigma$ ifadesi L dilinin grameri tarafından doğru oluşturulmuş bir katarı temsil eder,
  bir başka deyişle $L(G) = \{\sigma | n_0 \Rightarrow^* \sigma\}$.

Bağıntı

Bağıntılar bir yapıyı karakterize eder. önceki bölümde kümeleri incelemiştik. Bu bölümde kümelerin elemanları arasındaki küme içi bağıntılar ile temsil edilen bazı basit yapı formlarını inceleyeceğiz.

Küme içi bağıntılar

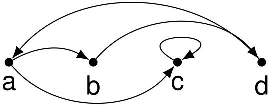
A kümesinde tanımlanmış küme içi bağıntı, kümedeki elemanların farklı sıralarda eşleşmeleri içeren topluluktur. Başka bir deyişle $R \subseteq A \times A$

Bağıntılar farklı şekillerde temsil edilebilir. örneğin $A = \{a, b, c, d\}$ kümesi üzerinde tanımlı α bağıntısı şu şekillerde temsil edilebilr.

1. Küme notasyonu: $\alpha = \{(a, b), (a, c), (b, d), (c, c), (d, a)\}$
2. Bağıntı matriksi:

$$\alpha = \begin{vmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{vmatrix}$$

3. Bağıntı çizgesi:



Bağıntı kuvveti

Bir α bağıntısının n'inci ($n \in \mathbb{N}$) kuvveti α^n olarak temsil edilen tümevarımla tanımlanmış bir kümedir.

i $\alpha^0 = \{(x,x) | x \in A\}$

ii $\alpha^{n+1} = \alpha^n \alpha \wedge n \in \mathbb{N}$

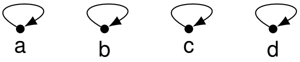
Teoremler

- $\alpha^m \alpha^n = \alpha^{m+n} \wedge (n,m \in \mathbb{N})$
- $(\alpha^m)^n = \alpha^{mn} \wedge (n,m \in \mathbb{N})$
- $\alpha^{-p} = (\alpha^{-1})^p = (\alpha^p)^{-1}$

Bu teoremler tümevarımla ispatlanabilir.

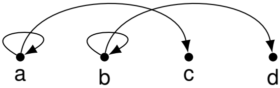
$< x,y > \in \alpha^n \Rightarrow$ x ve y arasında bağıntı çizgesinde en az bir yol bulunur. $A = \{a,b,c,d\}$ üzerinde tanımlı aşağıdaki bağıntıları inceleyelim

α^0 :

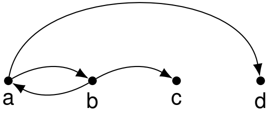


α^1 :

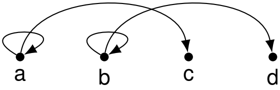
α^2 :



α^3 :



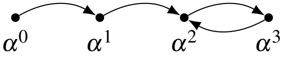
α^4 :



$\alpha^2 = \alpha^4 \qquad \alpha^4\alpha = \alpha^2\alpha = \alpha^3$

$\alpha^5 = \alpha^3 \qquad \alpha^6 = \alpha^4 = \alpha^2$

$\alpha^{2n+1} = \alpha^3 \qquad \alpha^{2n} = \alpha^2$ where $n \geq 1$



Teorem

A kümesi n elemanlı bir küme ve α A üzerine tanımlı bir bağıntı olsun. $[0, 2^{(n^2)}]$ aralığında $\alpha^s = \alpha^t$ eşitliğini sağlayan en az bir (s, t) çifti bulunur.

İspat.

$\alpha \subseteq A \times A$
$|A \times A|= n^2$, $|\mathscr{P}(A \times A)| = 2^{(n^2)}$ A üzerinde en fazla $2^{(n^2)}$ farklı bağıntı tanımlanabilir. öte yandan $[0, 2^{(n^2)}]$ aralığında α'nın $2^{(n^2)} + 1$ covet bulunur. Güvercin deliği ilkesine göre bu aralıkta en az bir eşdeğer bağıntı bulunmalıdır. $\square$

Teorem

α A kümesi üzerinde tanımlanmış bir bağıntı olsun. $\alpha^s = \alpha^t$ $s < t$ ve $p = t - s$

a $\alpha^{s+k} = \alpha^{t+k}; \forall k \geq 0$

b $\alpha^{s+kp+i} = \alpha^{s+i}; \forall k > 0 \wedge \forall i (0 < i < p)$

c If $\Sigma = \{\alpha^0, \alpha^1, \ldots, \alpha^{t-1}\}$ ise $\forall q \{\alpha^q \in \Sigma \wedge q \geq 0 \wedge q \in \mathbb{N}\}$

a ve b tümevarım ile ispatlanabilir.

c İspatı.

Eğer $q < t$ ise $\alpha^q \in \Sigma$
Eğer $q \geq t$ ise b'ye göre $q = s + kp + i$ $(i < p)$.
Böylece $\alpha^q = \alpha^{s+i}$
$i < p \Rightarrow s + i < t \Rightarrow \alpha^q \in \Sigma$ $\square$

Kapanış işlemleri

Kapalı kümeler

Bir küme, bir işlem küme elemanları üzerinde uygulandığında işlem sonucu yine aynı küme elemanlarına dahil ise o işlem için kapalıdır denir.

Kapanış işlemi

Bir X kümesi üzerinde tanımlanan bir işlem için, X'in S isimli bir alt kümesi üzerinde tanımlanabilen kapanış işlemi C(S), tanımlanan işlem için kapalı olan ve S'i içeren en küçük X alt kümesini tanımlar.

Kapanış işlemleri

Yansımalı Kapanış

α bağıntısı A kümesi üzerine tanımlı olsun. Yansımalı kapanış olan α' aşağıdaki özelliklere sahip olmalıdır

1. α' yansımalıdır
2. $\alpha \subseteq \alpha'$
3. Eğer $\alpha \subseteq \alpha''$ ve α'' yansımalı ise $\alpha' \subseteq \alpha''$

Bakışlı Kapanış

α bağıntısı A kümesi üzerine tanımlı olsun. Bakışlı kapanış olan α' aşağıdaki özelliklere sahip olmalıdır

1. α' bakışlıdır
2. $\alpha \subseteq \alpha'$
3. Eğer $\alpha \subseteq \alpha''$ ve α'' bakışlı ise $\alpha' \subseteq \alpha''$

Kapanış işlemleri

Geçişli Kapanış

α bağıntısı A kümesi üzerine tanımlı olsun. Geçişli kapanış olan α' aşağıdaki özelliklere sahip olmalıdır

1. α' geçişlidir
2. $\alpha \subseteq \alpha'$
3. Eğer $\alpha \subseteq \alpha''$ ve α'' geçişli ise $\alpha' \subseteq \alpha''$

- Yansımalı kapanış $r(\alpha)$ olarak ifade edilir
- Bakışlı kapanış $s(\alpha)$ olarak ifade edilir
- Geçişli kapanış $t(\alpha)$ olarak ifade edilir

E, A kümesi üzerinde tanımlı birim bağıntıyı ifade etsin.
$E = \{< x,x > | x \in A\}, \forall R \subseteq A \times A (E = R^0)$

Teorem

$r(\alpha) = R_\alpha \cup E$[1]

örneğin:

1. $r(R(<)) = R(\leq)$ ($<$ bir doğrusal sıradır ve $\leq$ bir tam sıradır)
2. $r(E) = E$
3. $r(R(\neq)) = A \times A$ evrensel bağıntı.

Ayrıca $r(R(\emptyset)) = E$

[1] yönlü bir çizgede her düğüm en fazla bir tane döngü içerebilir

Teorem

Eğer α bakışlı ise$\alpha = \alpha^{-1}$

Teorem

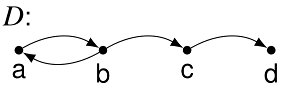
$s(\alpha) = \alpha \cup \alpha^{-1}$
Yönlü çizgedeki her ayrıt çift yönlü hale gelir.

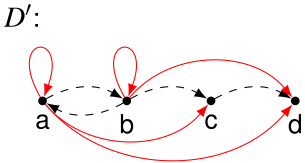
For instance:

1. $s(R(<)) = R(\neq)$
2. $s(R(\leq)) = A \times A$
3. $s(E) = E$
4. $s(R(\neq)) = R(\neq)$

Geçişli kapanış

D çizgesi α bağıntısına ait olsun. $t(\alpha)$'nın çizgesi D', D'de arasında bir yol bulunan her çifti birbirine bağlayan bir ayrıt eklenerek elde edilebilir.





Teorem

α, A kümesi üzerinde tanımlanmış bir bağıntı olsun.
$t(\alpha) = \bigcup_{i=1}^{\infty} \alpha^i = \alpha \cup \alpha^2 \cup \alpha^3 \cup \ldots$

İspat

İspatlamamız gereken $t(\alpha) \subseteq \bigcup_{i=1}^{\infty} \alpha^i$ ve $\bigcup_{i=1}^{\infty} \alpha^i \subseteq t(\alpha)$.

1. $\bigcup_{i=1}^{\infty} \alpha^i \subseteq t(\alpha)$ başka bir şekilde ifade etmek gerekirse
   $\forall < a,b > \in \alpha^i \Rightarrow < a,b > \in t(\alpha)$
   Tümevarımla ispatlanabilir.
   Taban durum tanım itibariyle geçerlidir $\alpha \subseteq t(\alpha)$
   Tümevarım adımı, varsayımımız $\alpha^n \subseteq t(\alpha)$
   $\forall < a,b > \in \alpha^{n+1}$ yeniden yazılabilir
   $\exists c, < a,c > \in \alpha^n \wedge < c,b > \in \alpha$
   $< a,c > \in t(\alpha)$ çünkü $< a,c > \in \alpha^n \subseteq t(\alpha)$
   $< c,b > \in t(\alpha)$ çünkü $< c,b > \in \alpha \subseteq t(\alpha)$
   $< a,b > \in t(\alpha)$ çünkü $t(\alpha)$ geçişlidir
   Böylece $\forall i (\alpha^i \subseteq t(\alpha))$ ve $\bigcup_{i=1}^{\infty} \alpha^i \subseteq t(\alpha)$

Teorem

α, A kümesi üzerinde tanımlanmış bir bağıntı olsun.
$t(\alpha) = \bigcup_{i=1}^{\infty} \alpha^i = \alpha \cup \alpha^2 \cup \alpha^3 \cup \ldots$

İspat.

İspatlamamız gereken $t(\alpha) \subseteq \bigcup_{i=1}^{\infty} \alpha^i$ ve $\bigcup_{i=1}^{\infty} \alpha^i \subseteq t(\alpha)$.

- b $t(\alpha) \subseteq \bigcup_{i=1}^{\infty} \alpha^i$
  $< a, b > \in \alpha^s$ ve $< b, c > \in \alpha^t$, $\bigcup_{i=1}^{\infty} \alpha^i$ içinde iki çift olsun.
  $< a, c > \in \alpha^{s+t}$ olduğundan $< a, c > \in \bigcup_{i=1}^{\infty} \alpha^i$
  Böylece $\bigcup_{i=1}^{\infty} \alpha^i$ geçişli olur ve tanım gereği $t(\alpha)$, α içeren tüm geçişli bağıntıların alt kümesidir.
  $t(\alpha) \subseteq \bigcup_{i=1}^{\infty} \alpha^i$

□

Teorem

α, A kümesi üzerinde tanımlanmış bir bağıntı olsun.
$t(\alpha) = \bigcup_{i=1}^{\infty} \alpha^i = \alpha \cup \alpha^2 \cup \alpha^3 \cup \ldots$

Bazı örnekler

a $b = a + 1 \ \forall(a,b) \in \mathbb{Z}$ bağıntısı için, geçişli kapanış $R(<)$'dir.

b α, A isimli bir tamsayı alt kümesi üzerinde tanımlı $R(<)$ bağıntısı olsun. A kümesini $R(<)$ bağıntısını kullanarak sıralamak için, $R(<) = t(\alpha')$'yı sağlayan en küçük α' bağıntısı bulunmalıdır . Eğer A kümesi sonlu ise $t(\alpha) = \bigcup_{i=1}^{2^{(n^2)}} \alpha^i$ olduğunu biliyoruz

$t(\alpha)$'yı oluşturma yöntemini veren teorem

Teorem

α n elemanlı bir A kümesi üzerinde tanımlı olsun.
$t(\alpha) = \bigcup_{i=1}^{n} \alpha^i$

İspat.

$\forall k(k > 0 \wedge \alpha^k \subset \bigcup_{i=1}^{n} \alpha^i)$ olduğunu göstermek yeterlidir
$\forall < x, y > \in \alpha^k$ ifadesi α'ya ait D çizgesinde arasında k uzunluklu yol bulunan tüm düğüm çiftlerini temsil eder.
$< x, y > \in \alpha^i$, $i = n$ ifadesi içinde geçişli bir bağıntı oluşturan tüm düğümleri içeren bir yolu temsil eder. $\square$

Teoremler

a Eğer α yansımalı ise $s(\alpha)$ ve $t(\alpha)$ de yansımalıdır.

b Eğer α bakışlı ise $r(\alpha)$ ve $t(\alpha)$ de bakışlıdır.

c Eğer α geçişli ise $r(\alpha)$ de geçişlidir ama $s(\alpha)$ geçişli olmayabilir.

a'nın İspatı.

Eğer α yansımalı ise $\alpha = E \cup \alpha'$

$s(\alpha) = E \cup \alpha' \cup E^{-1} \cup (\alpha')^{-1} = E \cup \alpha' \cup (\alpha')^{-1} = E \cup r(\alpha')$

$t(\alpha) = \bigcup_{i=1}^{n} (E \cup \alpha')^{i}$ ve $E\alpha' = \alpha' E = \alpha' \wedge E^{n} = E$

$t(\alpha) = E \cup \alpha' \cup (\alpha')^{2} \cup \ldots \cup (\alpha')^{n}$ yansımalıdır. $\square$

Teoremler

a $rs(\alpha) = sr(\alpha)$

b $rt(\alpha) = tr(\alpha)$

c $st(\alpha) \subseteq ts(\alpha)$

İspatlar.

a $sr(R) = s(R \cup E) = (R \cup E) \cup (R \cup E)^{-1} = R \cup E \cup R^{-1} \cup E^{-1}$
Hatırlayalım $E \cup E^{-1} = E$ ve $R \cup R^{-1} = s(R)$
$sr(R) = (R \cup R^{-1}) \cup E = s(R) \cup E = rs(R)$

$\square$

Teoremler

a $rs(\alpha) = sr(\alpha)$

b $rt(\alpha) = tr(\alpha)$

c $st(\alpha) \subseteq ts(\alpha)$

İspatlar.

b $tr(R) = t(R \cup E) \wedge rt(R) = t(R) \cup E$

$ER = RE \wedge E^n = E$

$(R \cup E)^n = E \bigcup_{i=1}^{n} R$

$$\begin{aligned} tr(R) = t(R \cup E) &= R \cup E \cup (R \cup E)^2 \cup \ldots \cup (R \cup E)^n \cup \ldots \\ &= E \cup R \cup R^2 \cup \ldots \cup R^n \cup \ldots \\ &= E \cup t(R) = rt(R) \end{aligned}$$

□

Teoremler

a $rs(\alpha) = sr(\alpha)$

b $rt(\alpha) = tr(\alpha)$

c $st(\alpha) \subseteq ts(\alpha)$

İspatlar.

c $R_2 \subseteq R_1 \Rightarrow s(R_2) \subseteq s(R_1) \wedge t(R_2) \subseteq t(R_1)$
$R \subseteq s(R) \wedge R \subseteq t(R)$ olduğu için
$t(R) \subseteq ts(R) \wedge s(R) \subseteq st(R)$
Eğer $s(R)$ bakışlı ise $t(s(R))$ de bakışlıdır.
Böylece $st(s(R)) = ts(R) \Rightarrow st(R) \subseteq ts(R)$.

$\square$

α, A üzerine tanımlı bir bağıntı olsun. $t(\alpha)$, α^+ şeklinde ve $tr(\alpha)$, α^* şeklinde ifade edilir. Bu tanımlar genellikle biçimsel dillerde, hesaplama modellerinde ve derleyici tasarımında kullanılır.
örneğin:
$P = \{P_1, P_2, \ldots, P_n\}$ ir program kitaplığında program ve alt programlar kümesi olsun. Bu kümede $\hookrightarrow$ ikili küme içi bağıntısı P_i programı P_j'yi çağırır anlamında kullanılsın.
Bu durumda $\hookrightarrow^+$ bağıntısı bir programın icrası sırasında çağrılan bir tüm alt programları kümesini ortaya çıkarmaya yarar. örneğin ($P_i \hookrightarrow^+ P_j$) P_i'nin icrası sırasında P_j çağrılır demektir.
Oysa $\hookrightarrow^*$ bir programın yürütülmesi sırasında herhangi bir anda etkin olacak olan program yada alt programlar kümesini ortaya çıkarmaya yarar.
$P_i \hookrightarrow^* P_j$, P_i'nin icrası sırasında P_j uyarılıyorsa $\forall i(P_i \hookrightarrow P_i)$ her zaman doğrudur ancak $P_i \hookrightarrow^+ P_i$ ise=> P_i özyinelidir (kendi kendini çağırıyordur).

Biçimsel Gramer

Biçimsel gramerler bir biçimsel dildeki katarların oluşturulmasını tanımlayan kurallar bütünüdür. Bu kurallar, dilin alfabesinin kullanılarak dildeki katarların dilin sözdizimine uygun olarak nasıl oluşturulabileceğini tanımlarlar.

Örneğin:
Aritmetik işlemleri oluşturmak için aşağıdaki kurallar kullanılabilir:
$\mathbb{Z} \quad + \quad - \quad \times \quad / \quad (\quad)$. Sıfıra bölüm hariç bütün doğru tanımlanmış aritmetik ifadeler anlamlıdır.
$(((2-1)/3)+4\times 6)$ doğru oluşturulmuş ve anlamlı bir aritmetik ifadedir. $2+(3/(5-(10/2)))$ da doğru oluşturulmuştur fakat sıfıra bölüm nedeniyle anlamlı değildir.

Gramer sözdizimi

Biçimsel gramerlerin sınıflandırılması ilk defa 1950li yıllarda Noam Chomsky tarafından önerilmiştir. Buna göre bir cümle yapısı grameri (phrase structure grammar) G biçimsel olarak $(N, \Sigma, n_0, \mapsto)$ çoklusuyla ifade edilir:

- N *uç olmayan ara simgeler* kümesi.
- Σ bir *uç simgeler* kümesi. N ile ayrık bir kümedir.
- n_0 başlangıç simgesi.
- $\mapsto$ üretim kuralları.
  $V = N \cup \Sigma$, $\mapsto$ bağıntısı V^* üzerine tanımlanmış olsun.
  $V^*NV^* \to V^*$ yapısına sahiptir.
  Burada $*$ Kleene yıldızı işlecidir.

Gramerlerin semantik yapısı

Bir gramerin işleyişi katarlar üzerinde tanımlı bağıntılar ile temsil edilebilir:

- Doğrudan türetme bağıntısı $G = (N, \Sigma, n_0, \mapsto)$ gramerinde V^*, $\Rightarrow_G$ olarak gösterilir ve şöyle tanımlanır:
  $x \Rightarrow_G y \leftrightarrow \exists l, w, r \in V^* : (x = lwr) \wedge (w \rightarrow w' \in \mapsto) \wedge (y = lw'r)$
- $\Rightarrow_G$'un yansımalı geçişli kapanışı bize erişilebilirlik bağıntısını $\Rightarrow^*{}_G$ verir.
  Doğrudan türetimin erişilebilirliği sonlu adımlı bir şekilde türetilebilecek sözcükleri verir.
  Bu nedenle $\sigma \in \Sigma^*$ in $n_0 \Rightarrow^*{}_G \sigma$ ifadesi L dilinin grameri tarafından doğru oluşturulmuş bir katarı temsil eder,
  bir başka deyişle $L(G) = \{\sigma | n_0 \Rightarrow^* \sigma\}$.

örnek 1

$S =$Harry,Sally,runs,swims,fast,often,long

$N =$sentecte,verbal sentence,noun,verb,adverb

$n_0 =$sentence

Gramer:

sentence $\mapsto$ noun verbal sentence

noun $\mapsto$ Harry

noun $\mapsto$ Sally

verbal sentence $\mapsto$ verb adverb

adverb $\mapsto$ fast

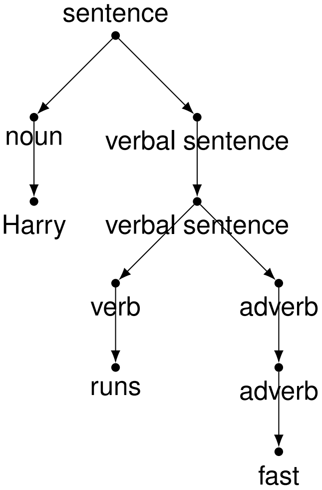
adverb $\mapsto$ often

adverb $\mapsto$ long

verb $\mapsto$ runs

verb $\mapsto$ swims

Bu kuralları iki farklı şekilde uygulayarak doğru bir biçimde cümle oluşturabiliriz. Bir seçenek hep en soldaki ifadeyi genişletmektir



$S =$Harry,Sally,runs,swims,fast,often,long

$N =$sentence,verbal

sentence,noun,verb,adverb

$n_0 =$sentence

Grammar:

sentence $\mapsto$ noun, verbal sentence

noun $\mapsto$ Harry

noun $\mapsto$ Sally

verbal sentence $\mapsto$ verb, adverb

adverb $\mapsto$ fast

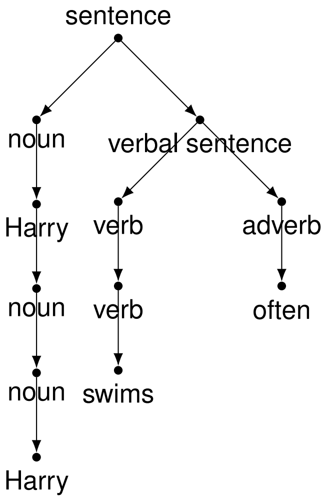
adverb $\mapsto$ often

adverb $\mapsto$ long

verb $\mapsto$ runs

verb $\mapsto$ swims

Bu kuralları iki farklı şekilde uygulayarak doğru bir biçimde cümle oluşturabiliriz. Diğer seçenek hep en sağdaki ifadeyi genişletmektir



S =Harry,Sally,runs,swims,fast,often,long
N =sentence,verbal
sentence,noun,verb,adverb
n_0 =sentence
Grammar:
sentence $\mapsto$ noun, verbal sentence
noun $\mapsto$ Harry
noun $\mapsto$ Sally
verbal sentence $\mapsto$ verb, adverb
adverb $\mapsto$ fast
adverb $\mapsto$ often
adverb $\mapsto$ long
verb $\mapsto$ runs
verb $\mapsto$ swims

Ayrıştırma

Ayrıştırma ya da sözdizimsel analiz, bir seri tekilden(kelimeler gibi) oluşmuş bir metnin sözdizimsel yapısının belirlenmesi amacıyla analiz edilmesidir.

Örnek 2

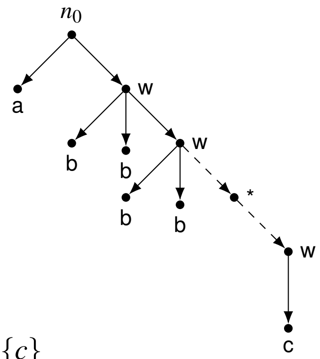
$S = a, b, c$

$N = n_0, w$

$\mapsto = \{n_0 \rightarrow aw | w \rightarrow bbw | w \rightarrow c\}$

örnek 2

$S = a, b, c$
$N = n_0, w$
$\mapsto = \{n_0 \to aw | w \to bbw | w \to c\}$



$L(G) = \{a\} \cdot \{bb\}^* \cdot \{c\}$
$= a(bb)^* c$
$\{a(bb)^n c | n \in \mathbb{N}\}$

örnek 3

$S = a, b, c$
$N = n_0, w$
$\mapsto = \{n_0 \rightarrow an_0b | n_0b \rightarrow bw | abw \rightarrow c\}$

$$\begin{aligned} n_0 &\Rightarrow an_0b \\ an_0b &\Rightarrow a(an_0b)b \Rightarrow a^n n_0 b^n \\ a^n(n_0b)b^{n-1} &\Rightarrow a^n bwb^{n-1} \\ a^{n-1}(abw)b^{n-1} &\Rightarrow a^{n-1}cb^{n-1} \end{aligned}$$

Ya da daha genel biçimiyle: $L(G) = \{a^m cb^m | m \in \mathbb{N}\}$

Example 4

$S = a, b, c$

$N = n_0, A, B, C$

$\mapsto = \{$

$n_0 \to A$

$A \to aABC$

$A \to abC$

$CB \to BC$

$bB \to bb$

$bC \to bc$

$cC \to cc$

$\}$

n_0

$\underline{A}$

$a\underline{A}BC$

$aa\underline{A}BCBC$

$aaab\underline{CB}CBC$

$aaab\underline{b}C\underline{CB}C$

$aaabb\underline{C}\underline{B}CC$

$aaab\underline{bB}CCC$

$aaabb\underline{bC}CC$

$aaabbb\underline{cC}C$

$aaabbbc\underline{cC}$

$aaabbbccc$

$a^3b^3c^3$

$L(G) = \{a^k b^k c^k | k \in \mathbb{N}^+\}$

Chomsky Hiyerarşisi

Chomsky hiyerarşisi biçimsel dillerin sınıflandırılması amacıyla oluşturulmuştur. Bir biçimsel dil $G = \{N, \Sigma, n_0, \mapsto\}$

Tip 0 grammar

Tip 0 gramer (kısıtsız gramer) bütün gramerleri içerir.
Turing makinesi ile tanınabilen tüm dilleri kapsar.
örnek 3 bir Tip 0 gramere uyar.

Chomsky Hiyerarşisi

Chomsky hiyerarşisi biçimsel dillerin sınıflandırılması amacıyla oluşturulmuştur. Bir biçimsel dil $G = \{N, \Sigma, n_0, \mapsto\}$

Tip 1 gramer

Tip 1 gramer (bağlama duyarlı(context sensitive) gramer) $w_1 \rightarrow w_2$ ise $|w_1| \leq |w_2|$ kuralına uygundur.
Örneğin bağlama bağımlılık $lwr \rightarrow lw'r$ şeklindeki bir kuralla açıklanabilir
Bu gramerler tarafından tanımlanan diller doğrusal sınırlandırılmış(linear bounded) otomatlar tarafından tanınabilir.
örnek 4 Tip 1 gramere aittir.

Chomsky Hiyerarşisi

Chomsky hiyerarşisi biçimsel dillerin sınıflandırılması amacıyla oluşturulmuştur. Bir biçimsel dil $G = \{N, \Sigma, n_0, \mapsto\}$

Tip 2 grammar

Tip 2 gramer (bağlamdan bağımsız(context free) gramer) bağlamdan bağımsız dilleri tanımlar.
Bu gramerin kuralları $A \rightarrow \gamma$ şeklindedir. A: Uç olmayan bir simge ve γ: bir katara karşılık gelir.
Bu gramerler tarafından tanımlanan diller yığın yapılı otomatlar tarafından tanınabilir
Bağlamdan bağımsız diller birçok programlama dilinin teorik temelini oluşturur.
örnek 1 Tip 2 gramere aittir.

Chomsky Hiyerarşisi

Tip 3 grammer

Tip 3 gramer (düzenli gramer) düzenli dilleri tanımlar.
Bu gramerde sol yan terminal olmayan tek simge, sağda bir kaç terminal sınıfı ya da bir kaç terminal simge ve en sağda tek bir terminal olmayan simge bulunur.
Bu diller sonlu durumlu otomatlar tarafından tanınabilir.
Ayrıca, bu tip diller düzenli ifadelerle de tanımlanabilirler.
Düzenli diller arama örüntüleri tanımlamada sıklıkla kullanılır.
örnek 2 Tip 3 gramere aittir.

Sadece Tip 2 ve Tip 3 gramerler sözdizim ağaçlarına sahiptir.
$n_0 \rightarrow \Lambda$ kuralı boş sözcükleri desteklemek için herhangi bir gramere katılabilir.
Tip 3 $\subseteq$ Tip 2 $\subseteq$ Tip 1 $\subseteq$ Tip 0

Chomsky Hiyerarşisi

Tipi	Dil (Gramer)	Gramerdeki türetim kuralları	Tanıyan Otomat
0	Recursively enumerable (kısıtsız)	$\alpha \to \beta$ $(\alpha, \beta \in (N \cup \Sigma)^*,$ α değişken içerir$)$	Turing makinesi
1	Bağlama duyarlı	$\alpha \to \beta$ $(\alpha, \beta \in (N \cup \Sigma)^*, \lvert\beta\rvert \geq \lvert\alpha\rvert,$ α değişken içerir$)$	doğrusal sınırlı otomat
2	Bağlamdan bağımsız	$A \to \alpha$ $(A \in N, \alpha \in (N \cup \Sigma)^*)$	yığın yapılı otomat
3	Düzenli	$A \to aB$ $(A, B \in N, a \in \Sigma)$	Sonlu otomat

Backus-Naur Form

BNF

BNF notasyonu gramer tanımlamakta kullanılabilecek bir biçimdir. BNF sözdiziminde

- $\mapsto$ ::= ile ifade edilir
- uç olmayan simgeler < > içerisinde yazılır
- uç simgeler olduğu gibi yazılır
- tekrarlar | işareti ile ayrılır

Backus-Naur Form

Tip 0 Gramer

$$
\begin{aligned}
< n_0 > &::= a < n_0 > b \\
< n_0 > b &::= b < w > \\
ab < w > &::= c
\end{aligned}
$$

Backus-Naur Form

Tip 1 Gramer

$$
\begin{aligned}
< n_0 > &::= < A > \\
< A > &::= a < A >< B >< C > |ab < C > \\
< C >< B > &::= < B >< C > \\
b < B > &::= bb \\
b < C > &::= bc \\
c < C > &::= cc
\end{aligned}
$$

Backus-Naur Form

Tip 2 Gramer

$$
\begin{aligned}
< \text{sentence} > &::=< \text{noun} >< \text{verbal sentence} > \\
< \text{noun} > &::= \text{Harry} | \text{Sally} \\
< \text{verbal sentence} > &::=< \text{verb} >< \text{adverb} > \\
< \text{verb} > &::= \text{runs} | \text{swims} \\
< \text{adverb} > &::= \text{fast} | \text{often} | \text{long}
\end{aligned}
$$

Backus-Naur Form

Tip 3 Gramer

$$< n_0 > ::= a < w >$$
$$< w > ::= bb < w > | c$$

$w \rightarrow bbw$ şeklindeki kurallara özyineli kurallar denir. Eğer bir özyineli kuralda uç olmayan bir simge en sağda ise buna normal üretim kuralı denir.

Düzenli İfade

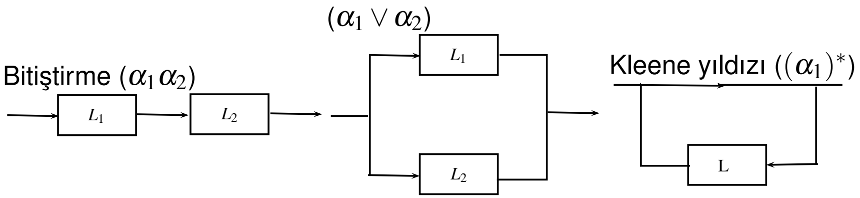
Σ üzerinde düzenli ifadeler tümevarımla tanımlanır:

1. Λ ve Σ'nın her elemanı bir düzenli ifadedir
   $L(\Lambda) = \{\Lambda\}; L(a) = \{a\}; \forall a \in \Sigma$
2. Bitiştirme işlemi($\cdot$) iki düzenli ifade üzerine uygulandığında yeni bir düzenli ifade elde edilir.
   $L(\alpha\beta) = L(\alpha)L(\beta)$
3. ($\vee$) işlemi tiki düzenli ifade üzerine uygulandığında yeni bir düzenli ifade elde edilir.
   $L(\alpha \vee \beta) = L(\alpha) \vee L(\beta)$
4. Kleene yıldızı işlemi($*$) iki düzenli ifade üzerine uygulandığında yeni bir düzenli ifade elde edilir.
   $L(\alpha^*) = L^*(\alpha)$

Teorem

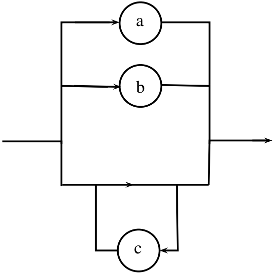
L, S kümesi üzerine tanımlanmış bir dil olsun. $L \subseteq S^*$. L'nin düzenli bir dil olması için bir düzenli gramer G'ye uygun olarak üretilmiş olması gerekir. Diğer bir deyişle $L = L(G)$.
Düzenli ifadeler düzenli diller tanımlar. Dil ile düzenli ifade arasında bir izomorfizm vardır. Yapısal denklik demek olan izomorfizm mühendislikte modellemede sıklıkla kullanılır.

Sözdizim çizgeleri düzenli ifade tanımlamada kullanılabilir. Aşağıdaki üç çizge düzenli ifadeler üzerinde yapılabilecek temel işlemleri gösterir.

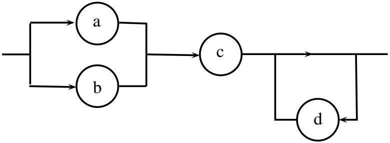


Sözdizim çizgeleri üzerine örnekler:

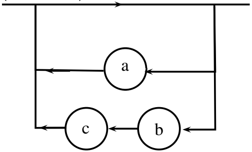
$a \vee b \vee c^*$



$(a \vee b)cd^*$



$(a \vee bc)^*$



Düzenli ifadeler üzerinde pratik yapmak için Eclipse Graphrex[2] eklentisi kullanılabilir.

[2]Web sitesi: http://crotonresearch.com/graphrex/