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 Autotamata, 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

Düzenli diller

Düzenli diller

- Düzenli diller dile ait semboller değiştiği zaman hala düzenli midir? Temelde bu soru kapalılık (closure) özelliği ile irdelenebilir.
- Kapalılık özelliği birçok farklı özellikteki dilin sınırlarını belirlemek için önemli bir araçtır.
- Bir dil kararsız sonlu durum otomatı veya düzenli bir gramer ile belirtilebiliyor ise o zaman bu dil düzenlidir.
- Peki başka bir yöntem ile bir dilin düzenli olup olmadığı bulunabilir mi? Evet.
- Kapalılık özelliği ile bir dilden kapalılığı bozmayacak şekilde türetilen her dil düzenli olacaktır.
- Bunları irdelemek için düzenli dillerin nasıl davrandığının araştırmamız gereklidir.

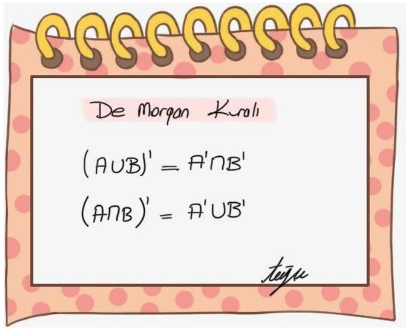
Düzenli diller

Teorem 4.1

- Eğer $L1$ ve $L2$ düzenli diller ise o zaman $L1 \cup L2$, $L1 \cap L2$, $L1L2$, $L1^{-1}$, and $L1^*$ da düzenli bir dildir.
- Bunu kanıtlamak için önce L1 ve L2 dillerini kabul eden düzenli ifadelerin birleşimlerinin de düzenli bir ifade olduğunu göstermemiz gereklidir. Bu bir önceki bölümde Genel Geçiş Şeması (GTG) bölümünde gördüğümüz birleşimlerle yapılabilir.
- Bu durumda
  - *Birleşim:* $L1 \cup L2$
  - Ardışıklı: $L1L2$
  - Çoklama: $L1^*$
  - Tersi: $L1^{-1}$
- Yukarıdaki tüm durumlarda oluşan düzenli gramerler hemen görülebilir ki düzenlidir.

Düzenli Gramerler

- Teorem 4.1'e göre
- L1 ve L2 kesişimi aşağıdaki şekilde birleşimin tersi şeklinde ifade edilebilir.



- Bu durumda kesişim işlemi kapalılık özelliğini sağlar. Nasıl?
- Düzenli iki dilin tersi işlemlerimim birleşiminin tersi düzenlidir. Bu ifadeye erişmek için $L1^{-1}$ (tersi) işlemi bir dilin tersinin düzenli olup olmadığını bilmemiz gerekir. Önceki sunumlarda dil için tersi işleminin kapalılık özelliğine uyduğunu görmüştük. Bu durumda;
- Tersi işlemin de dil düzenli ise oluşan dil yine düzenli olur. O zaman birleşimin tersi de düzenlidir ve bu durumda kesişim işlemi de düzenli olacaktır.

Homo morphism

- Bir dil için S ve L iki farklı alfabe olsun. Bu durumda S alfabesini L alfabesine dönüştüren h fonksiyonu aşağıdaki gibidir.

h: S -> L

- h fonksiyonun işlevi homomorphism olarak isimlendirilir. Aşağıda w dilinin homorfik imajı h(w) verilmiştir.

$$w = a_1 a_2 \cdots a_n,$$

$$h(w) = h(a_1)\, h(a_2) \cdots h(a_n)$$

Homo morphism

- Yanda homorofik imaja örnek verilmiştir.
- Homomorfik imaj yanda h ile verilmiştir.

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

$L = \{aa, aba\}$

$$\downarrow$$

$$\begin{aligned} h(a) &= ab, \\ h(b) &= bbc. \end{aligned}$$

$$\downarrow$$

$h(L) = \{abab, abbbcab\}$

Homomorphism

- Eğer bir dile ait semboller h fonksiyonu ile değiştiriliyorsa dilin özellikleri korunur.
- Bu durumda dil düzenli bir dil ise oluşan yeni dilde yine düzenli olur.
- Yanda ∑ ve Γ ile verilen iki alfabe için h homorfik fonksiyonu verilmiştir. Buna göre r düzenli dili için oluşan h(r) dili düzenlidir.

Take $\Sigma = \{a, b\}$ and $\Gamma = \{b, c, d\}$. Define h by

$$h(a) = dbcc,$$
$$h(b) = bdc.$$

If L is the regular language denoted by

$$r = (a + b^*)(aa)^*,$$

then

$$r_1 = \left(dbcc + (bdc)^*\right)(dbccdbcc)^*$$

denotes the regular language $h(L)$.

L1/L2

- L1 ve L2 için L1/L2 ön ek operatörü L2 + L1 şeklinde tanımlanır. Örneğin;

$$L_1 = L\left(a^*baa^*\right),$$

$$L_2 = L\left(ab^*\right).$$

- Örneğin; L2 + L1 = a*b + a*baa* olacaktır. Bunu sadeleştirdiğimizde a*ba* elde edilir.

Düzenli diller

- Aşağıda belirtilen dil düzenli midir? İspatlayınız?

$$L = \{a^n b^n : n \geq 0\}$$

Kanıtlama (Pigeonhole)

- Pigeonhole yöntemine göre n adet objeyi m adet kutuya koymamız gerekiyor. Burada n > m ise o zaman en az bir kutuda 1'den fazla nesne olmalıdır. Bunu kullanarak $L = \{a^n b^n : n \geq 0\}$ ifadesinin düzenli olmadığını kantılayabiliriz.
- İlk etapda bu dilin düzenli olduğunu kabul edelim. Bu durumda i=1...n'e kadar başlangıç durumundan n adet a'yı yakalamak için aşağıdaki d geçişi mevcut olmalıdır.

$$\delta^* (q_0, a^n) = q$$

- Bu işlem döngü kurmadan n'adet a yı yaklamak için benzer şekile farklı bir sayı n'den farklı bir sayı olan m adet durum kullanmamızı gerekli kılacaktır.

$$\delta^* (q_0, a^m) = q,$$

Kanıtlama (Pigeonhole)

- Burada $L = \{a^n b^n : n \geq 0\}$ L dilini kabul etmek için b'lerin de yakalanması gereklidir. Aşağıda q_f son durumu belirtmektedir.

$$\delta^* (q, b^n) = q_f \in F.$$

- m ve n birbirinden farklı sayılar olduğu durumlar için aşağıdaki geçişler sağlanabilir.

$$\begin{aligned} \delta^* (q_0, a^m b^n) &= \delta^* (\delta^* (q_0, a^m), b^n) \\ &= \delta^* (q, b^n) \\ &= q_f. \end{aligned}$$

- Bu durumda başta belirttiğimiz m=n için belirtilen geçişler düzenli olma şartını sağlayacaktır. Ama n ≠ m durumunda kabul ettiğimiz dil için şartlar sağlanamaz, çelişki oluştuğu için bu dil düzenli olamaz.

Pumping Lema

- Bir önceki örnekten yola çıkarak aşağıdaki şartlarda düzenli bir dil oluştuğunu varsayabiliriz.
  - Eğer döngü yoksa o zaman dil sonludur ve dolayısıyla düzenlidir.
  - Eğer döngüler varsa ve bunların hiçbiri boş durum geçişi barındırmıyor ise bu dil karasız sonlu durum (nfa) olarak ifade edilebilir.
  - Eğer arada döngüler varsa ve bu döngüler için geçişlerin sayısının bir önemi yoksa dil düzenlidir. Örneğin; w_1 w_2 , w_1 **v** w_2 , ve w_1 **v v** w_2 ...
  - Eğer m adet sayıda durum için döngüye girme aşamasında m adet sembol geçişi sağlanmışsa dil düzenlidir. Örneğin m 3 olsun. Bu durumda w_1 w_2 w_3 v, aaa bbbbbb, abbcccccc...

Pumping Lema

- Eğer yeterince uzun bir karakter katarını kabul eden bir dil varsa bu dili yakalayan kararsız sonlu durum otomatı 3 ana kısıma ayrılabilir. Bu kısımlar başlangıç, orta, ve son şeklindedir. Burada ortadaki kısım tekrarlı yada döngü barındıran kısımdır. Bu kısımda kabul edilen dil de baştaki dilin bir alt kümesidir ve bu kısım (pumped) yığınlaşmıştır. Burada yığınlanan kısım içinde bu tanımın geçerli olduğunu kabul edebiliriz.
- m sayıda duruma sahip L dili için; ve w $w = xyz$ karakter katarı için;
- xy uzunluğu m'den küçüktür: $|xy| \leq m,$
- y uzunluğu birden büyüktür: $|y| \geq 1,$
- Bu tanıma erişilir: $w_i = xy^i z,$

Pumping Lemma

- Eğer L düzenli ise, onu tanıyan bir DFA (Karalı Sonlu Durum Otomatı) bulunur. Böyle bir DFA'nın q0, q1, q2, ..., qn etiketli durumları olduğunu varsayalım. Şimdi L dilindeki $|w| \geq m = n+1$ olan bir w dizesini alalım. L dilinin sonsuz olduğu varsayıldığı için, bu şart her zaman mümkün olabilir. Otomat tarafından w dizesini işlerken geçtiği durumların kümesini düşünelim, diyelim ki

  q0, qi, qj , ..., qf .

- Bu sıra tam olarak $|w| + 1$ girişe sahip olduğundan, en az bir durum tekrarlanmalıdır ve böyle bir tekrar, en geç n'inci adımda başlamalıdır. Bu nedenle, sıra şu şekilde görünmelidir: q0, qi, qj, ..., qr, ..., qr, ..., qf, bu da gösterir ki w'nin x, y ve z alt dizeleri olmalıdır, böylece

  $\delta^*(q0, x) = qr,$

  $\delta^*(qr, y) = qr$

  $\delta^*(qr, z) = qf$

- Ve $|x y| \leq n + 1 = m$ ve $|y| \geq 1$. Bu durumdan hemen şu sonuç çıkar:

  $\delta^*(q0, xz) = qf$

  $\delta^*(q0, xy^2z) = qf$

  $\delta^*(q0, xy^3z) = qf$

Pumping Lemma - Örnek

- Önceki örneği ele alıcak olursak:
- Pumping Lemma kullanarak, $L = \{a^n b^n : n \geq 0\}$ dilinin düzenli olmadığını gösterin. L'nin düzenli olduğunu varsayalım, bu nedenle pumping lemmanın geçerli olması gerekir. M'nin değerini bilmiyoruz, ancak ne olursa olsun her zaman n = m'yi seçebiliriz. Bu nedenle, alt dizi y'nin tamamen 'a' harflerinden oluşması gerekir. |y| = k varsayalım. Daha sonra $w_i = xy^i z,$ için i = 0 kullanarak elde edilen dize şu şekildedir:
- $w_0 = a^{m-k} b^m$ ve açıkça L içinde değildir. Bu, pumping lemmaya aykırıdır ve bu, L'nin düzenli olduğu varsayımının yanlış olduğunu gösterir.

Pumping Lemma - Örnek

- Sözlük $\Sigma = \{a, b\}$ olsun. Dil $L = \{w \in \Sigma* : n_a (w) < n_b (w)\}$ düzenli değildir. Bir m değeri verildiğini varsayalım. w'yi seçme özgürlüğümüz olduğu için, $w = a^m b^{m+1}$ seçeriz. Şimdi pumping lemma ile tanımlanan $w = xyz$ içinde $|xy|$ m'den büyük olamaz, sadece a'lardan oluşan bir y seçebiliriz, yani $y = a^k$, $1 \leq k \leq m$. Şimdi $i = 2$ kullanarak pumping döngüsü kurabiliriz. Bu döngüde y kısmının içindeki a sayısı k kadar artmış olacaktır. Bizde $y=a^k$ olduğu için elde edilen yeni dize $w_2 = a^{m+k}b^{m+1}$ L içinde değildir. Çünkü k>=2 ve m+k < m+1 olamaz. Bu nedenle, pomping lemması ihlal edilir ve L düzenli değildir.

Pumping Lemma - Örnek

- **L = ((ab)^n a^k : n > k, k ≥ 0)** olan dil düzenli değildir. Pumping lemma: $w_i = xy^iz,$
- Verilen m değeri için, **w = (ab)$^{m+1}$ a^m** olan dizeyi seçiyoruz, bu da L dilindedir. **|xy| ≤ m** kısıtlaması nedeniyle, hem x hem de y ab harflerinden oluşan dize kısmında olmalıdır. x'in seçimi argümanı etkilemez, bu nedenle y ile ne yapılabilir görelim. Şimdi y'yi a olarak seçersek, *i = 0*'ı seçeriz ve ***L((ab)*a*)*** dilinde olmayan bir dize elde ederiz. Eğer y'yi ab olarak seçersek, tekrar i = 0'ı seçebiliriz. Şimdi (ab)^m a^m olan bir dize elde ederiz, bu da L dilinde değildir çünkü n>k şartı sağlanmaz. Aynı şekilde, başka yapabileceğimiz herhangi bir olası seçimle benzer şekilde çıkarım yapabiliriz, böylece iddiamızı kanıtlarız.

Pumping Lemma - Örnek

- Verilen dil $L = \{a^n$: n bir mükemmel kare$\}$ nin düzensiz olduğunu gösterin.

$w_i = xy^i z,$

- Rakibin seçtiği m için $w = a^{m^2}$'yi seçeriz.
- Eğer w = xyz ayrımı ise, açıkça $y = a^k$, ile $1 \le k \le m$. Bu durumda,
- $w_0 = a^{m^2-k}$
- Ancak, $m^2 - k > (m - 1)^2$ olduğundan, w0 L'de olamaz. Bu nedenle dil düzensizdir.