CENG 306 Biçimsel Diller ve Otomatlar
Formal Languages and Automata

PUSH DOWN AUTOMATA

Yığın Yapılı Otomatlar

Konular

- Pushdown Automata (PDA)
- Pushdown Automata and Context-Free Grammars

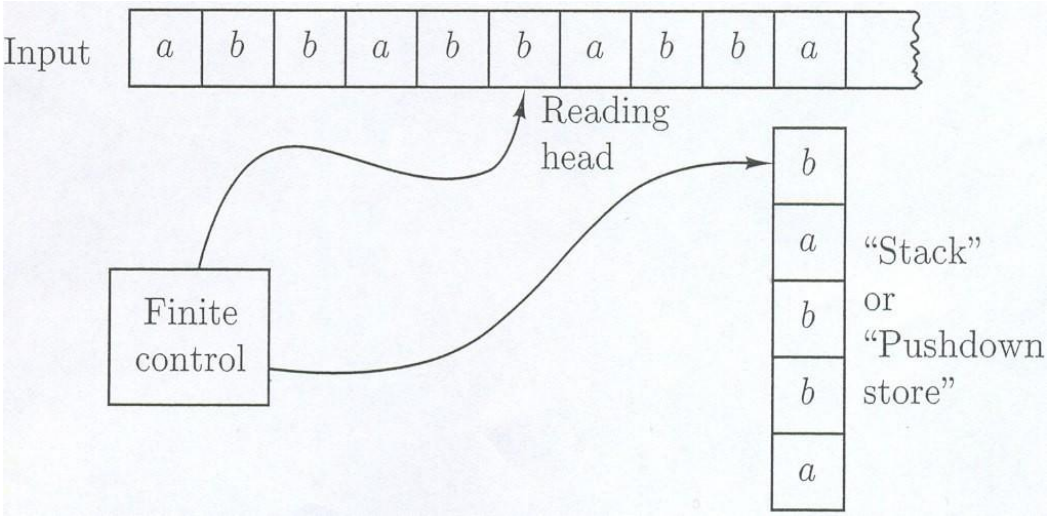
Pushdown Automata

- Regular olmayan context-free dillerin tanınması finite automata ile **yapılamaz**.
- $\{ww^R : w \in \{a, b\}^*\}$ dilini tanımak için fazladan ne gerekir?
- Bu dil
  - $S \rightarrow aSa$,
  - $S \rightarrow bSb$
  - $S \rightarrow e$

  kurallarına sahip bir grammar tarafından üretilebilir.
- Böyle bir dili tanıyan cihazın string'in yarısına geldiğinde ilk yarısını hafızada tutması ve ikinci yarısını bununla ters sırada karşılaştırması gerekir.
- Bu tür dillerin tanınmasında pushdown automata (PDA) kullanılır.

- Pushdown automata hafıza birimi olarak bir yığın (stack) kullanır.

Pushdown Automata



- Regular olmayan (nonregular) dillere diğer bir örnek dengelenmiş parantez üreten dildir.
- Pushdown automata bu dili tanırken stack count sıfır ile başlar.
- Her sol parantez gelişinde stack count 1 artar ve her sağ parantez gelişinde 1 azalır.
- String soldan sağa doğru okunurken negatif değere ulaşılması veya string bittiğinde pozitif değer olması kabul edilmeyen string'i gösterir.
- String **bittiğinde** stack count **sıfır** ise string kabul edilir.

Pushdown Automata

Tanım:

- *Pushdown automata $M = (K, \sum, \Gamma, \Delta, s, F)$ şeklinde bir altılı (6-tuple) ile tanımlanır.*

K	*durumlar*
$\sum$	*alfabe (giriş sembolleri için)*
Γ	*alfabe (stack sembolleri için)*
$s \in K$	*başlangıç durumu*
$F \subseteq K$	*sonuç durumları kümesi*
Δ	*geçiş ilişkisi* $(K \times (\sum \cup \{e\}) \times \Gamma^*) \times (K \times \Gamma^*)$

Pushdown Automata

((p, α , β), (q, γ)) ∈ Δ ise;

- *Pushdown automata p durumundadır.*
- *input tape'ten a okunmuştur (read). (a = e ise input tape'e başvurulmaz)*
- *Stack üzerinde en üstte β çekilerek (pop) γ ile degiştirilir (push).*
- *q durumuna geçilir.*
- *β = e ise stack'tan çekme yapılmaz.*
- *γ = e ise stack'a yazma yapılmaz.*

Pushdown Automata

- *Bu pushdown automata nondeterministic'tir.*
- *push stack'ın en üstüne sembol/semboller ekler,*
- *pop ise en üstteki sembolü/sembolleri alır.*
- *((p, u, e), (q, **a**))* ***a'yı push*** *yapar,* *((p, u, **a**), (q, e))* ***a'yı pop*** *yapar.*
- *Okunan string'in soldaki kısmı sonraki işlemler üzerinde etki yapmaz.*
- *Pushdown automata için configuration* $K \times \Sigma^* \times \Gamma^*$ *olarak tanımlanır.*
  - *K automata'nın bulunduğu durumu,*
  - Σ^* *input string'te okunmamış kısmı,*
  - Γ^* *ise stack'taki string'i gösterir.*
- *(q, w, abc)* *için stack'ta* ***en üstte a,*** *en altta c vardır.*

Pushdown Automata

(p, x, α) bir adım sonra (q, y, ζ) 'yı oluşturur ve

(p, x, α) ⊢ M (q, y, ζ) şeklinde gösterilir eğer;

- *((p, a, β), (q, γ)) ∈ Δ şeklinde bir ilişki varsa,*
- *ve x = ay, α = βμ, ve ζ = γμ, μ ∈ Γ* ise*

⊢M için reflexive, transitive, closure'u ⊢M şeklinde gösterilir.*

M pushdown automata'ı w∈ ∑ string'ini kabul eder* ***eğer***

(s, w, e) ⊢M*(p, e, e), p∈ F ise

Konfigürasyonlar $C_0 \vdash_M C_1 \vdash_M \ldots \vdash_M C_n$ şeklinde gösterilir.

Eger C_0 = (s, w, e) ve C_n = (p, e, e) ve p∈ F ise w string'i kabul edilir.

Pushdown Automata

Örnek: $L = \{wcw^R : w \in \{a, b\}^*\}$ dilini kabul eden bir pushdown automata oluşturalım.

(ababcbaba∈ L, abcab∉ L, cbc∉ L)

Yöntem: c gelene kadar herşeyi yığına at, c'den sonra her gelen sembol için yığının üstünde de aynı sembol varsa çek

Pushdown Automata

Örnek: $L = \{wcw^R : w \in \{a, b\}^*\}$ dilini kabul eden bir pushdown automata oluşturalım.

(ababcbaba∈ L, abcab∉ L, cbc∉ L)

$M = (K, \sum, \Gamma, \Delta, s, F)$, $K = \{s, f\}, \sum = \{a, b, c\}, \Gamma = \{a, b\}, F = \{f\}$

Δ toplam 5 adet geçiş ilişkisine sahip olsun;

1. *((s, a, e), (s, a))*
2. *((s, b, e), (s, b))*
3. *((s, c, e), (f, e))*
4. *((f, a, a), (f, e))*
5. *((f, b, b), (f, e))*

Otomat string'in ilk yarısını okurken (c'ye kadar) başlangıç durumunu korur ve input tape'ten okuduğunu push eder, c okuduktan sonra final state'e geçer ve input tape'ten okuduğuyla stack'tan okuduğunu karşılaştırır .

Nondeterministic pushdown automata'dır.

Pushdown Automata

Örnek: (devam) $L = \{wcw^R : w \in \{a, b\}^*\}$ dilini kabul eden bir pushdown automata oluşturalım. *abbcbba için geçişler aşağıdaki tabloda verilmiştir.*

1. $((s, a, e), (s, a))$
2. $((s, b, e), (s, b))$
3. $((s, c, e), (f, e))$
4. $((f, a, a), (f, e))$
5. $((f, b, b), (f, e))$

State	Unread Input	Stack	Transition Used
s	$abbcbba$	e	-
s	$bbcbba$	a	1
s	$bcbba$	ba	2
s	$cbba$	bba	2
f	bba	bba	3
f	ba	ba	5
f	a	a	5
f	e	e	4

Giriş string'i bittiğinde stack boş degilse, giriş string'i ile stack arasında farklı karakter okuma yapılırsa, giriş string'i bittiğinde ve/veya stack'ta okunacak sembol olmadığında sonuç durumunda (f) değil ise string kabul edilmez.

Pushdown Automata

Örnek: $L = \{w \in \{a, b\}^* : w$ *aynı sayıda a ve b'ye sahiptir.*$\}$ dilini kabul eden bir pushdown automata oluşturalım.

Yöntem: her a geldiğinde yığından b, her b geldiğinde a çekebilirim

final durumunu nasıl belirleyeceğim? Yığın sonu işaretçisi.

Pushdown Automata

Örnek: $L = \{w \in \{a, b\}^* : w$ *aynı sayıda a ve b'ye sahiptir.*$\}$ dilini kabul eden bir pushdown automata oluşturalım.

$M = (K, \sum, \Gamma, \Delta, s, F)$, $K = \{s, q, f\}$, $\sum = \{a, b\}$, $\Gamma = \{a, b, c\}$, $F = \{f\}$

Δ *toplam 8 adet geçiş ilişkisine sahip olsun;*

1. $((s, e, e), (q, c))$ *c stack'ın sonunu göstermek için kullanılıyor.*
2. $((q, a, c), (q, ac))$
3. $((q, a, a), (q, aa))$
4. $((q, a, b), (q, e))$
5. $((q, b, c), (q, bc))$
6. $((q, b, b), (q, bb))$
7. $((q, b, a), (q, e))$
8. $((q, e, c), (f, e))$

Otomat ilk önce yığına c yazar ve ara duruma (q) geçer.

Her a'ya karşılık b veya b'ye karşılık a geldiğinde yığından pop yapılır

diğer durumlarda input ile yığından pop edilen kaynaştırılarak yığına push edilir

$((K \times (\sum \cup \{e\}) \times \Gamma^*) \times (K \times \Gamma^*)$ *olduğundan POP ve PUSH için* Γ^* *olabilir.*

Pushdown Automata

Örnek: (devam) *L = {w ∈ {a, b}* : w aynı sayıda a ve b'ye sahiptir.}* dilini kabul eden bir pushdown automata oluşturalım.

abbbabaa için geçişler aşağıdaki tabloda verilmiştir.

State	Unread Input	Stack	Transition	Comments
s	$abbbabaa$	e	-	Initial configuration.
q	$abbbabaa$	c	1	Bottom marker.
q	$bbbabaa$	ac	2	Start a stack of a's.
q	$bbabaa$	c	7	Remove one a.
q	$babaa$	bc	5	Start a stack of b's.
q	$abaa$	bbc	6	
q	baa	bc	4	
q	aa	bbc	6	
q	a	bc	4	
q	e	c	4	
f	e	e	8	Accepts.

Pushdown Automata

Tanım:

Her finite automata *basit bir pushdown automata olarak görülebilir.*

M = (K, ∑ , Δ, s, F) bir nondeterministic finite automata ve

M' = (K, ∑ , φ, Δ', s, F) pushdown automata ve

Δ'= {((p, u, e), (q, e)) : (p, u, q) ∈ Δ,} *şeklinde tanımlanır.*

M' stack üzerine boş string yazar ve okuma yapmaz böylelikle finite automata'nın geçişlerini simule eder.

Pushdown Automata and CFGs

- Pushdown automata'lar context-free dilleri tanımak icin gerekli olan özelliklere sahiptir.
- PDA'lar özellikle context-free dil olan programlama dillerinin analizinde kullanılmaktadırlar.
- PDA'lar programlama dillerinde syntax analyzer olarak kullanılmaktadır.

Pushdown Automata and CFGs

Theorem: *Her **context-free dil** bir **pushdown otomat** tarafından kabul edilir.*

Proof : *$G = (V, \sum, R, S)$ bir CFG olsun. $L(M) = L(G)$ olacak şekilde bir pushdown otomat oluşturmak zorundayız.*

Bu otomatın iki durumu (p, q) olsun ve M stack alfabesi olarak terminal ve nonterminalleri (V) kullansın.

$M = (\{p, q\}, \sum, V, \Delta, p, \{q\})$

Pushdown Automata and CFGs

Proof : (devam)

M = ({p, q}, ∑, V, Δ, p, {q})

Δ toplam 3 adet geçiş ilişkisine sahip olsun;

1. *((p, e, e), (q, S))*
2. *((q, e, A), (q, x)) her bir $A \rightarrow x \in R$ için*
3. *((q, a, a), (q, e)) her $a \in \sum$ için*

PDA, G'nin başlangıç sembolü S'yi stack'a push ederek başlar ve q durumuna geçer.

Daha sonraki adımlarda stack'ın en üstündeki A sembolü ile x sembolünü degiştirir

($A \rightarrow x \in R$) veya girişten okunan sembol ile aynı olan stack'ın en üstündeki terminal sembolü pop eder.

Bu PDA leftmost derivation yapar. Nondeterministic çalışır.

Pushdown Automata and CFGs

Örnek: $G = (V, \sum, R, S)$, $V = \{S, a, b, c\}$, $\sum = \{a, b, c\}$, ve

$R = \{S \rightarrow aSa, S \rightarrow bSb, S \rightarrow c\}$ şeklinde tanımlı bir CFG olsun ve

$L = \{wcw^R : w \in \{a, b\}^*\}$

dilini oluştursun.

Bu dili tanıyan bir PDA olan $M = (\{p, q\}, \sum, V, \Delta, p, \{q\})$ olarak tanımlanabilir.

$\Delta = \{ ((p, e, e), (q, S)),$ (T1)
$((q, e, S), (q, aSa)),$ (T2)
$((q, e, S), (q, bSb)),$ (T3)
$((q, e, S), (q, c)),$ (T4)
$((q, a, a), (q, e)),$ (T5)
$((q, b, b), (q, e)),$ (T6)
$((q, c, c), (q, e))\}$ (T7)

Pushdown Automata and CFGs

Örnek: (devam)

abbcbba için geçişler aşagıdaki tabloda verilmiştir.

State	Unread Input	Stack	Transition Used
p	$abbcbba$	e	-
q	$abbcbba$	S	T1
q	$abbcbba$	aSa	T2
q	$bbcbba$	Sa	T5
q	$bbcbba$	$bSba$	T3
q	$bcbba$	Sba	T6
q	$bcbba$	$bSbba$	T3
q	$cbba$	$Sbba$	T6
q	$cbba$	$cbba$	T4
q	bba	bba	T7
q	ba	ba	T6
q	a	a	T6
q	e	e	T5

M = (K, Σ, Γ, Δ, s, F) PDA'sı aşağıdaki gibi verilmiştir:

K = {s, f},

F = {f},

Σ = {a, b},

Γ = {a},

Δ = {((s, a, ε), (s, a)), ((s, b, ε), (s, a)), ((s, a, ε), (f, ε)), ((f, a, a), (f, ε)), ((f, b, a), (f, ε))}.

(a) **aba** .girişi için M PDA'sındaki tüm konfigürasyon geçişlerini veriniz.

(b) **aba, aa, abb** $\notin$ L(M) ve **baa, bab, baaaa** $\in$ L(M) olduğunu gösteriniz.

(c) Sözel olarak bu makinanın yaptığı işi tanımlayınız.

L(M)={w $\in$ Σ* | }

M = (K, Σ, Γ, Δ, s, F) PDA'sı aşağıdaki gibi verilmiştir:

K = {s, f}, F = {f}, Σ = {a, b}, Γ = {a},

Δ = {((s, a, ε), (s, a)), ((s, b, ε), (s, a)), ((s, a, ε), (f, ε)), ((f, a, a), (f, ε)), ((f, b, a), (f, ε))}.

(a) **aba** .girişi için M PDA'sındaki tüm konfigürasyon geçişlerini veriniz.

(b) **aba, aa, abb** $\notin$ L(M) ve **baa, bab, baaaa** $\in$ L(M) olduğunu gösteriniz.

(c) Sözel olarak bu makinanın yaptığı işi tanımlayınız.

L(M)={w $\in$ Σ* |

(a)

(s, aba, ε) $\vdash$ (s, ba, a) $\vdash$ (s, a, aa) $\vdash$ (s, ε, aaa)

(s, aba, ε) $\vdash$ (s, ba, a) $\vdash$ (s, a, aa) $\vdash$ (f, ε, aa)

(s, aba, ε) $\vdash$ (f, ba, ε)

Bu konfigürasyonlardan hiçbiri kabul edilebilir bir konfigürasyonla sonuçlanmıyor: aba $\notin$ L(M) .

(b) a'daki gibi yapılabilir.

(c) L(M)={w $\in$ Σ* | w katarının tam ortasında her zaman a sembol vardır}

L(M) = {xay $\in$ Σ* : |x| = |y|}.

Construct pushdown automata that accept the following:

$L = \{a^m b^n : \mathbf{m \leq n \leq 2m}\}$.

a'yı saymak için yığını kullanmamız ve daha sonra bu sayıyı okuduğumuz gibi b'lerle karşılaştırmamız gerekmektedir. Buradaki komplikasyon, her a için bir veya iki b olabilir. Yani nondeterminisme ihtiyacımız olacak. Unutmayın ki önce a'lar sonra ise b'ler geliyor.

Her bir a gördüğümüzde, bunun bir b ile mi yoksa iki b ile eşleşeceğini bilmiyoruz.

The first is to **push either one or two characters onto the stack for a's**. **In this case, once we get to the b's, we'll pop one character for every b we see.** A nondeterministic machine that follows all paths of combinations of **one or two pushed characters** will find at least one match for every string in L.

Birincisi, a'lar için bir veya iki sembolü yığınlar için yığının üzerine itmektir. Bu durumda, b'lere ulaştığımızda, gördüğümüz her b için bir karakter çekmemiz gerekir. Bir veya iki itilen karakterin tüm kombinasyon yollarını takip eden non-deterministik bir makine, L dilindeki her katar için en az bir eşleşme bulacaktır.

Alternatif olarak, her a için tek bir karakter itmek (push) ve b'leri işlerken belirsizliği sağlamaktır: Her yığın karakteri için bir b veya iki b kabul ediyoruz.

İşte ikinci yaklaşımı benimseyen bir PDA. Bunu başka bir yolla yazmayı da deneyebilirsiniz. Bu makinenin aslında üç duruma ihtiyacı var: iki b'nin okunduğu ancak sadece tek bir a'nın açıldığı duruma izin vermek için b'leri işlemek için iki duruma ihtiyaç duyar. $M = (\{s, f, g\}, \{a, b\}, \{a\}, \Delta, s, \{f, g\})$, burada

$\Delta = \{$ $((s, a, \varepsilon), (s, a))$, /* Read an a and push one onto the stack */

$((s, \varepsilon, \varepsilon), (f, \varepsilon))$, /* Jump to the b reading state */

$((f, b, a), (f, \varepsilon))$, /* Read a single b and pop an a */

$((f, b, a), (g, \varepsilon))$, /* Read a single b and pop an a but get ready to read a second one */

$((g, b, \varepsilon), (f, \varepsilon))\}$. /* Read a single b without popping an a */

Aşağıda verilen L dilini ele alalım:
L = {$w^R w''$ | $w \in \{a, b\}^*$ ve w'' *w katarındaki* her *a* sembolünün *b* ile ve *b* sembolünün *a* ile değiştiği katarı gösterir}.

L(M) sağlayan M PDA'sını elde ediniz.

aaabbb $\in$ L(M) bbbbaaaa $\in$ L(M) babbaaba $\in$ L(M) abaababbab $\in$ L(M)

Aşağıda verilen L dilini ele alalım:
L = {$w^R w''$ | $w \in \{a, b\}^*$ ve w'' *w katarındaki* her *a* sembolünün *b* ile ve *b* sembolünün *a* ile değiştiği katarı gösterir}.

L(M) sağlayan M PDA'sını elde ediniz.

aaabbb $\in$ L(M) bbbbaaaa $\in$ L(M) babbaaba $\in$ L(M) abaababbab $\in$ L(M)

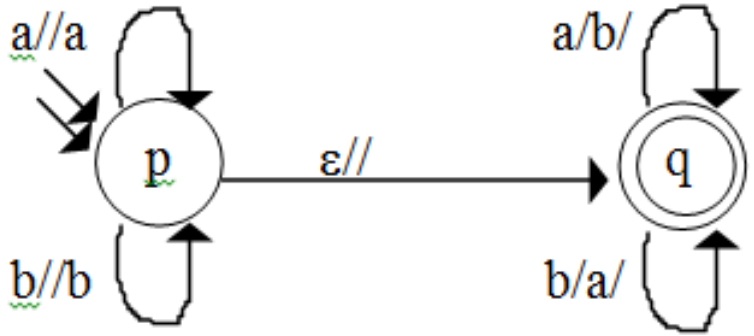
Yöntem: Her a ve b geldiğinde yığına atarım, sonra nondeterministik olarak bir andan itibaren her a geldiğinde b, her b geldiğinde a çekerim

M = ({p,q}, {a, b}, {a}, Δ, p, q), burada

Δ = { ((p, a, ε), (p, a)),
((p, b, ε), (p, b)),
((p, ε , ε), (q, ε)),
((q, a, b), (q, e)),
((q, b, a), (q, e))
}

M = ({p,q}, {a, b}, {a}, Δ, p, q), burada

Δ = { ((p, a, ε), (p, a)),
((p, b, ε), (p, b)),
((p, ε , ε), (q, ε)),
((q, a, b), (q, e)),
((q, b, a), (q, e))
}



Ödev

- Problemleri çözünüz 3.3.1a, 3.3.1b (sayfa 135)
- Problemleri çözünüz 3.3.2a, 3.3.2b, 3.3.2c, 3.3.2d (sayfa 135)
- Problemleri çözünüz 3.4.1 (sayfa 142)