

Biçimsel Diller ve Otomata Teorisi

Hafta 12:
İçerikten Bağımsız Diller (II. Bölüm)

Plan

1. Pushdown Otomata (PDA) Giriş
2. PDA Geçişler
3. PDA Örnekler
4. PDA'nun Formal Gösterimi
5. JFLAP ile PDA
6. PDA ile İçerikten Bağımsız Dillerin Denkliği

Pushdown Otomata (PDA) Giriş

Pushdown kelimesinin karşılığı olarak aşağı itimli kelimesini düşünebiliriz.

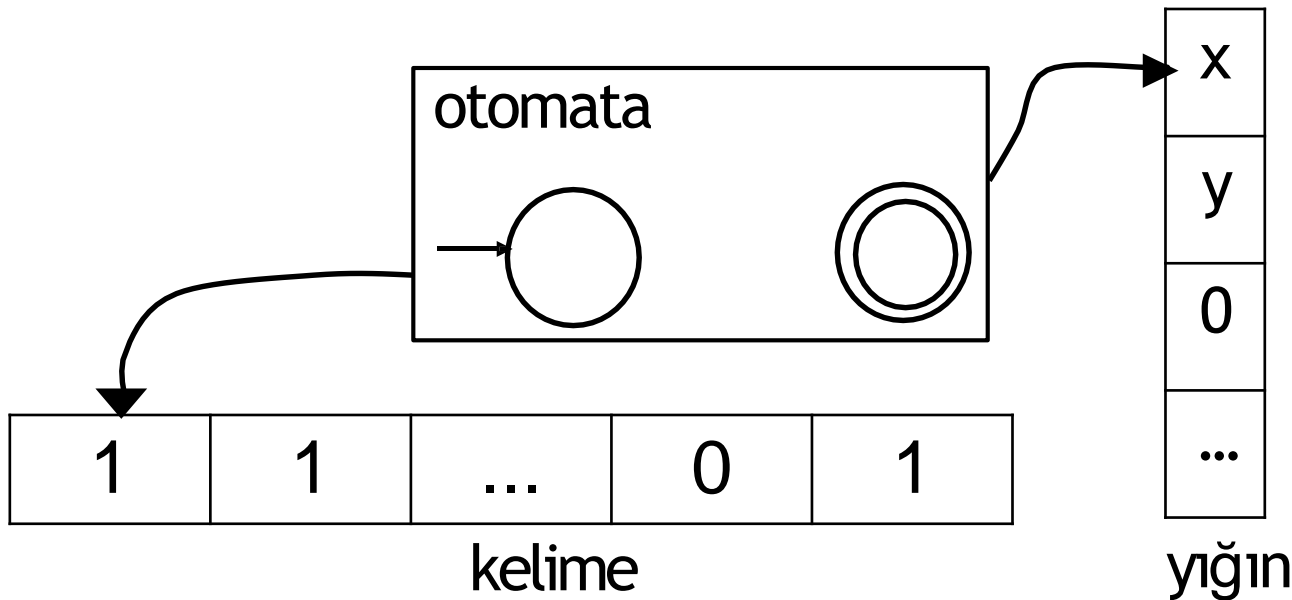
Pushdown otomata aslında bir nondeterministik sonlu otomata (NFA) dır. Konvensiyonel NFA'dan farkı bir **yığına** (stack) sahip olmasıdır.

Bu yığını restoranlarda tabakların üst üste konulduğu ve altında bu tabakaları yukarıya doğru iten bir mekanizmanın olduğu alete benzetebiliriz.

PDA'lar içerikten bağımsız gramerlerin gücüne sahiptirler. İçerikten bağımsız gramerler içerikten bağımsız dilleri üretirler, PDA'lar ise içerikten bağımsız dilleri tanırlar.

Pushdown Otomata (PDA) Giriş

Yandaki resimdeki alete tabaklar en usten birer birer konur, ve yeni tabaklar geldikce eski tabaklar asagi dogru itilir. Buradan bir tabak almak istedigimizde en usttekini aliriz. Bunun sonucunda bir alttaki tabak en uste cikar. Bu kuralin adi son gelen ilk giderdir (Last In First Out).



Pushdown Otomata (PDA) Giriş

PDA harfleri yığına yazabilir \Rightarrow Yazılan harf yığının en üstüne konur, yığındaki diğer harfler aşağı itilir.

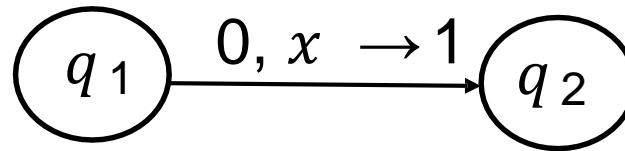
PDA yığından harfler okuyabilir \Rightarrow Okunan harf yığının en üsteki harfidir, bu harf okunduğunda yığından çıkartılmış olur, böylece yığındaki diğer harfler bir basamak üste çıkar.

Yığının sonuz miktarda bilgi tutabildiği varsayılır. Yani yığına istediğimiz kadar harf koyabiliriz.

PDA'nin ikinci en önemli özelliği nondeterministik olmasıdır. Böylece aynı anda birden fazla durum aktif olabilir.

Pushdown Otomata Geçişler

En geniş anlamda, bir PDA'da bir durumdan diğer bir duruma, o anki durum, okunan kelimenin harfi ve yığının en üstteki harfi dikkate alınarak geçilir.



Yukarıdaki örnekte, q_1 durumunda iken eğer kelimenin sıradaki harfi 0 ve yığının en üstteki harfi x ise q_2 'ye geçeriz.

Burada x harfini yığından okuruz, yani böylece bu harfi yığından çıkarmış oluruz.

Daha sonra yığının en üstüne 1 harfini koyarız. (Yığından x' i alır, yerine 1 koyarız).

Pushdown Otomata Geçişler

Epsilondan Kaynaklanan Bazı Özel Haller:

1. $\varepsilon, x \rightarrow 1$ ise okunan kelimenin harfi ne olursa olsun yığının en üstünde x harfi varsa q_1 'den q_2 'ye geçiş sağlanır. Yığından x çıkartılır. 1 yığına eklenir (en üstüne konur).
2. $0, \varepsilon \rightarrow 1$ ise yığına bakmaksızın 0 harfini okuduğumuzda geçiş yaparız ve 1 harfini yığının en üstüne koyarız (yığından bir şey çıkartılmaz).
3. $0, x \rightarrow \varepsilon$ ise okunan kelimedeki sıradaki harf 0 ise ve yığının en üstündeki harf x ise geçiş sağlanır. Bu geçiş sonucunda yığından (en üstünden) x harfi çıkartılır. Yığına yeni bir harf konmaz.
4. $\varepsilon, \varepsilon \rightarrow \varepsilon$ ise okunan kelimeye ve yığına bakılmaksızın geçiş olur. Bu geçiş sonucunda yığından bir harf okunmaz (yığından bir harf çıkmaz) ve yığına bir harf konmaz.

Pushdown Otomata Örnek

$L = \{0^n 1^n \mid n \geq 1\}$ dilini tanıyan PDA tasarlıyalım.

Bu dilin düzenli olmadığını Pumping lemma yardımıyla görmüştük.

Bundan başka, 1 harflerini okumaya başlamadan önce daha önce kaç tane 0 harfi okuduğunu hatırlaması gerektiği içinde bu dili tanıyan bir NFA tanımlayamamıştık.

Not. PDA'larda \$ özel bir harftir (simgedir). Yığının en sonundaki (en dibindeki) harfi temsil eder. Bu harf ürettiğimiz yığınla birlikte gelmez. Bunu bilhassa yığına eklemek gerekir.

Bir yığındaki \$ harfini okuduğumuzda (yani yığından bu harfi çıkarttığımızda) yığının içinde hiçbir harf kalmaz. Yığın boşalmış olur.

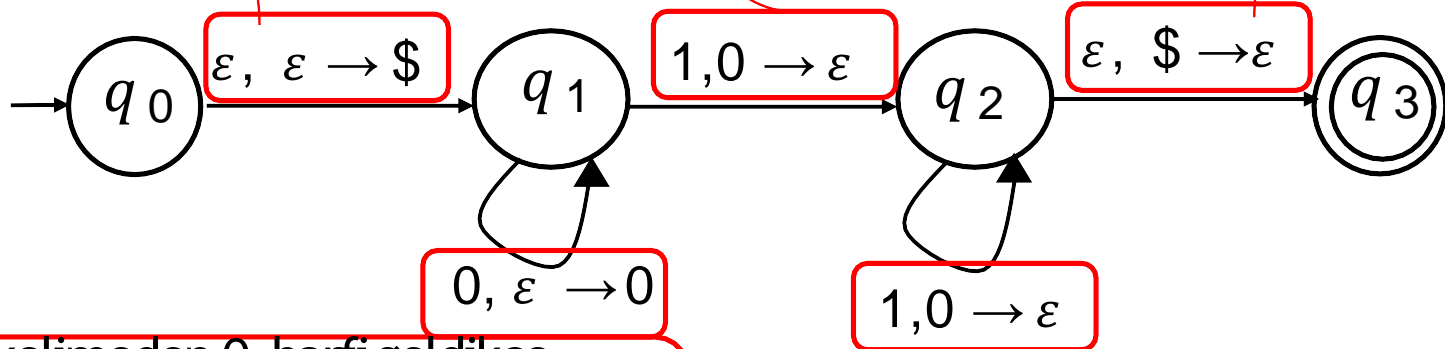
Not. Bazı kaynaklarda \$ yerine Z yada Z_0 da kullanılır.

Pushdown Otomata Örnek 1

q_0 'da harf okumaksizin ve yığına bakılmaksizin q_1 'e gidilir. \$ yığına eklenir.

q_2 'de iken eğer yığının en üstünde \$ varsa yani yığının en alt harfi varsa q_3 'e geçilir.

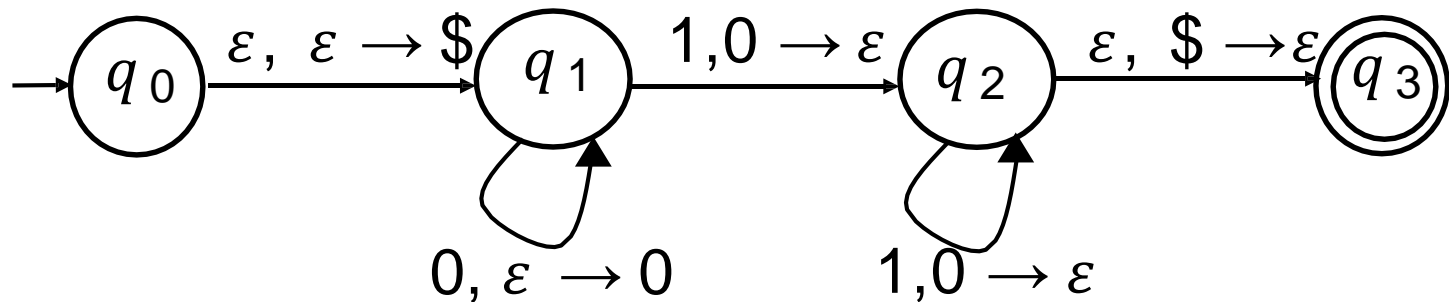
q_1 'de iken kelimeden 1 harfi gelirse yığından bir 0 okuyarak q_2 'ye geceriz



q_1 'de kelimeden 0 harfi geldikçe q_1 'de kalınır, ve her bir 0 harfi için bir 0 harfi yığına eklenir.

q_2 'de iken eğer kelimeden 1 harfi gelirse, bu 1'lere karşılık, yığından bir tane 0 okunur (yani çıkarılır) ve yığına bir şey eklenmez.

Pushdown Otomata Örnek 1

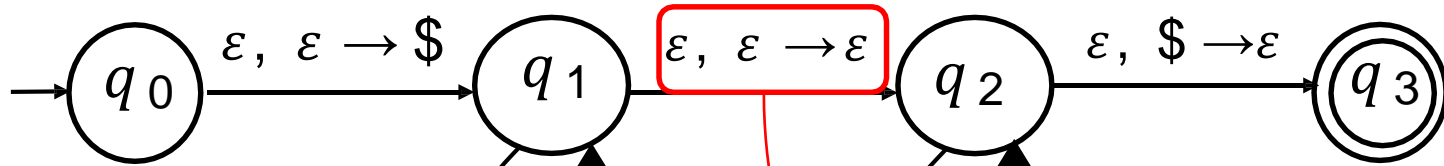


Yukarıdaki PDA elimizde varken $w = 0^2 1^2$ kelimesini okuyalım.

Okunan Harf	Aktif Durum	Yığın
	q_0, q_1	\$
0	q_1	0, \$
0	q_1	0,0, \$
1	q_2	0, \$
1	q_2	\$
	q_3	—

Pushdown Otomata Örnek II

$L = \{ w w^R \mid w \in \{0,1\}^* \}$ dilini tanıyan PDA tasarlıyalım. Burada w^R w kelimesinin tersten yazılmış halidir. Örneğin $w = 01$ iken $w^R = 10$ olur. Su halde L dili çift uzunluktaki palindromların dilidir.



q_1 'de kelimeden 0 harfi geldikçe q_1 'de kalınır, ve yığına 0 harfi eklenir.
 q_1 'de kelimeden 1 harfi geldikçe q_1 'de kalınır, ve yığına 1 harfi eklenir.

q_1 'de iken kelimeden ve yığından hiçbir harf okumadan ve yığına hiçbir şey eklemeden q_2 'ye geçilir.

q_2 'de iken kelimeden 0 (1) gelirse ve yığının en üstünde 0 (1) varsa, yığından 0 (1) çıkartılarak yine q_2 'de kalınır.

Pushdown Otomata Formal Gosterimi

Bir pushdown otomata 6-li siradir $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$. Burada

1. Q durumlarin kumesidir,
2. Σ alfabe (kelime icin),
3. Γ yiginin kullandigi alfabe,
4. $\delta: Q \times \Sigma_{\varepsilon} \times \Gamma_{\varepsilon} \rightarrow \mathcal{P}(Q \times \Gamma_{\varepsilon})$, ge
5. $q_0 \in Q$ baslangic durumu,
6. $F \subseteq Q$ final durumlarinin kumesidir.

Not 1. Yiginin kullandigi alfabe Γ (yigina ekledigimiz harfler) kelimenin alfabesinden (Σ) farkli olabilir. Ornegin $\$ \in \Gamma$.

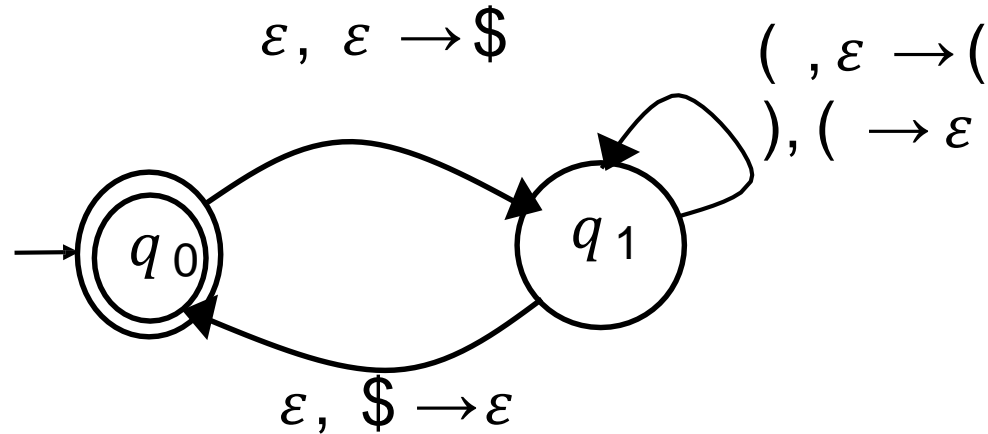
Not 2. Gecis fonksiyonun 3 tane girisi vardir: $Q, \Sigma_{\varepsilon}, \Gamma_{\varepsilon}$. Bu demektirki bir PDA'da, bir durumdan digerine gecebilmek icin, icinde bulunulan durum, kelimenin o anki harfi ve yiginin o andaki en ustteki harfi dikkate alinir. Gecis icin bunlarinin ucunun de c musait olmasi gerekir.

Pushdown Otomata Formal Gosterimi

Not 3. Gecis fonksiyonun deger kumesi $\mathcal{P}(Q \times \Gamma_\epsilon)$ dir. Gecis sonunda birden fazla duruma gecebilecegimizi ve birden fazla harfi yigina koyabilecegimizi ifade eder.

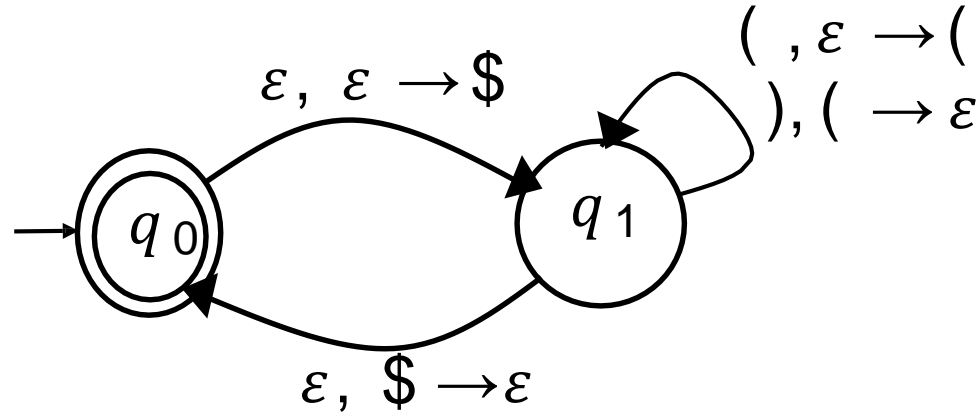
or. $\{(q_1, \emptyset) (q_2, \emptyset)\} \in \mathcal{P}(Q \times \Gamma_\epsilon)$.

or. $M = (\{q_0, q_1\}, \{(,)\}, \{\$, (,)\}, \delta, q_0, \{q_0\})$ PDA'sı şu şekilde olsun:



Bu PDA, düzgün olarak ic ice geçmiş parentezlerin dilini tanıyan PDA'dır.

Bu PDA için $w = (())()$ kelimesini okuyalım:

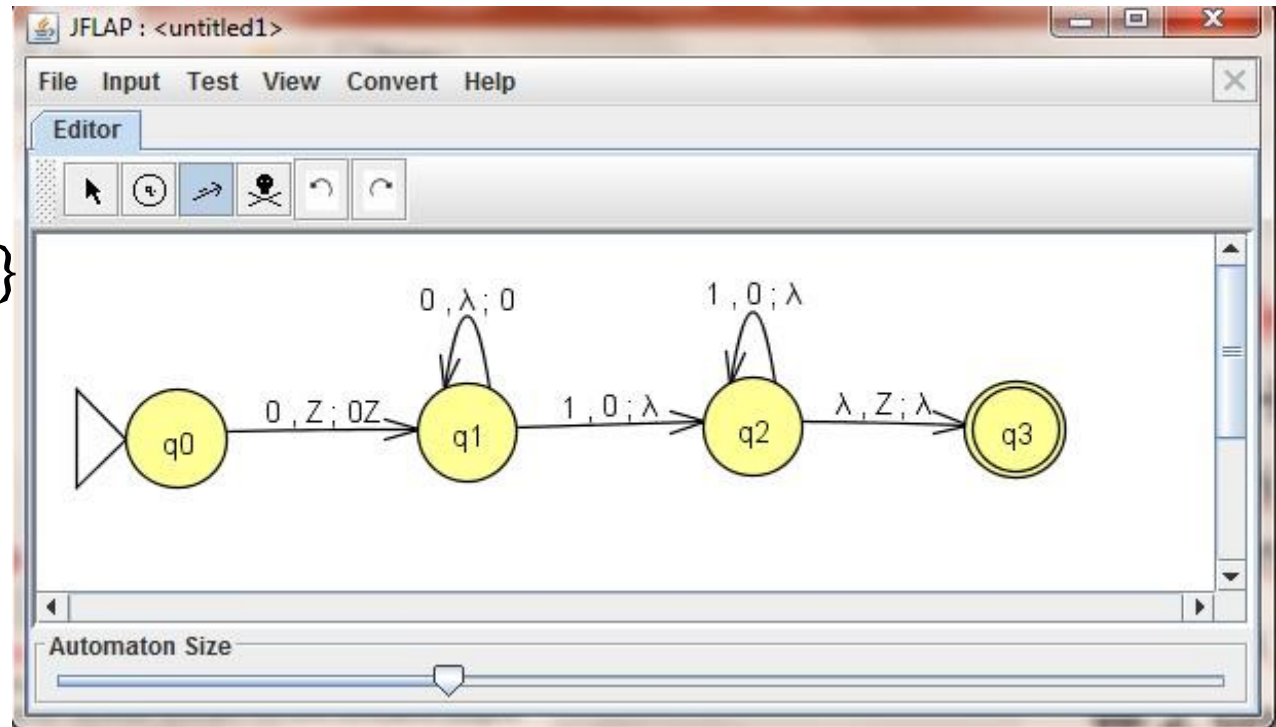


Okunan Harf	Aktif Durum	Yığın
	q_0, q_1	\$
(q_1	(, \$
(q_1	(, (, \$
)	q_1	(, \$
)	q_0	–
	q_0, q_1	\$
(q_1	(, \$
)	q_0	–

JFLAP ile Pushdown Otomata

JFLAP → Pushdown Automata → Multiple Character Input
adimleri izlendikten sonra daha önce sonlu otomatalarda yaptığımız
gibi, durumlar ve geçişler oluşturulur.

JLAP'da yiginin son elemanını simgelemek için \$ yerine Z kullanılır.
Ve Z simgesi, JFLAP'da bir PDA oluşturduğumuzda otomatik
olarak gelir.



$L = \{0^n 1^n \mid n \geq 1\}$
(ϵ yerine
 λ kullanılıyor.)

PDA ile İçerikten Bağımsız Dillerin Birbirine Denkliği

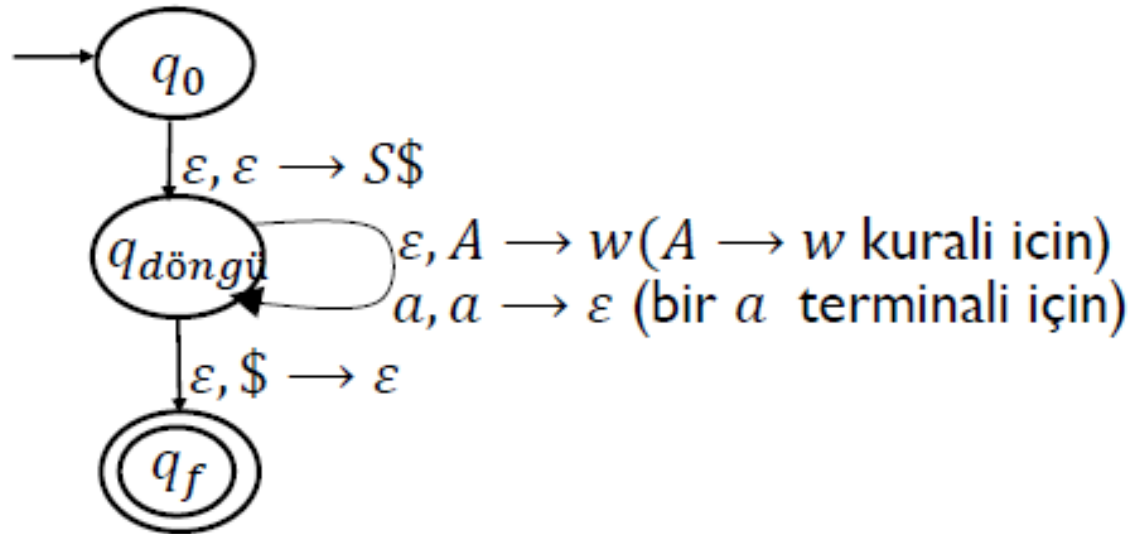
Teorem: Bir dilin içerikten bağımsız olması için gerek ve yeter şart o dilin bir pushdown otomata tarafından tanınmasıdır.

Lemma 1: Bir dil içerikten bağımsız ise onu tanıyan bir pushdown otomata vardır.

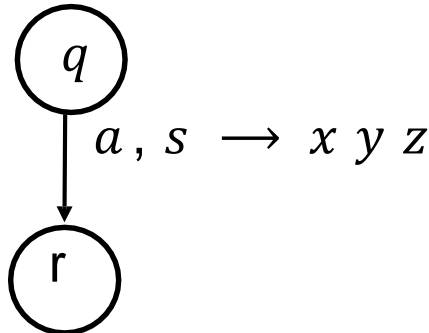
Bir içerikten bağımsız gramerden bir PDA elde ederek, verilen bir kelimenin bu gramerden üretilip üretilmediğini kolayca görebiliriz. Böylece bu iş için CYK algoritmasına alternatif bir yöntemimiz olur.

İçerikten Bağımsız Grammerlerden Pushdown Otomataya

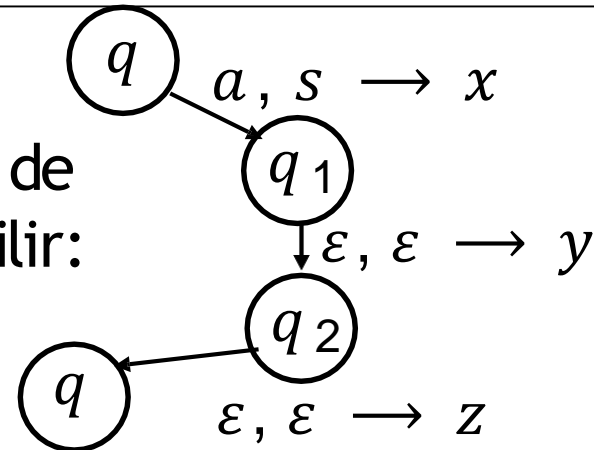
Bir içerikten bağımsız grammerden türetilen PDA'nın üç durumu vardır: q_0 , $q_{dönü}$ ve q_f . Bu PDA şu formdadır:



Not:



şu şekilde de gösterilebilir:



İçerikten Bağımsız Grammerlerden Pushdown Otomataya

Bir içerikten bağımsız grammerden PDA türetmek için aşağıdaki prosedür izlenir.

1. Bir önceki şekildeki gibi q_0 , $q_{dönüş}$ ve q_f durumları çizilerek birbirine bağlanır.
2. Yığına başlangıç değişkeni S ve $\$$ eklenir. ($\varepsilon, \varepsilon \rightarrow S\$$)
3. Değişkenler (nonterminaller) yigindan çıkartılarak yerlerine bu değişkene karşılık gelen kelime eklenir. Örneğin bir $A \rightarrow w$ kuralı için yigindan A çıkartılıp w kelimesi en sağ sembolden başlayarak eklenir. Ortaya çıkan geçişler $q_{dönüş}$ 'ye eklenir.
4. Yigindan terminaller çıkartılır. Örneğin bir a terminali için, $a, a \rightarrow \varepsilon$ geçisi $q_{dönüş}$ 'ye eklenir.
5. $\$$ yigindan çıkartılarak ($\varepsilon, \$ \rightarrow \varepsilon$) final durumu q_f 'ye gidilir.

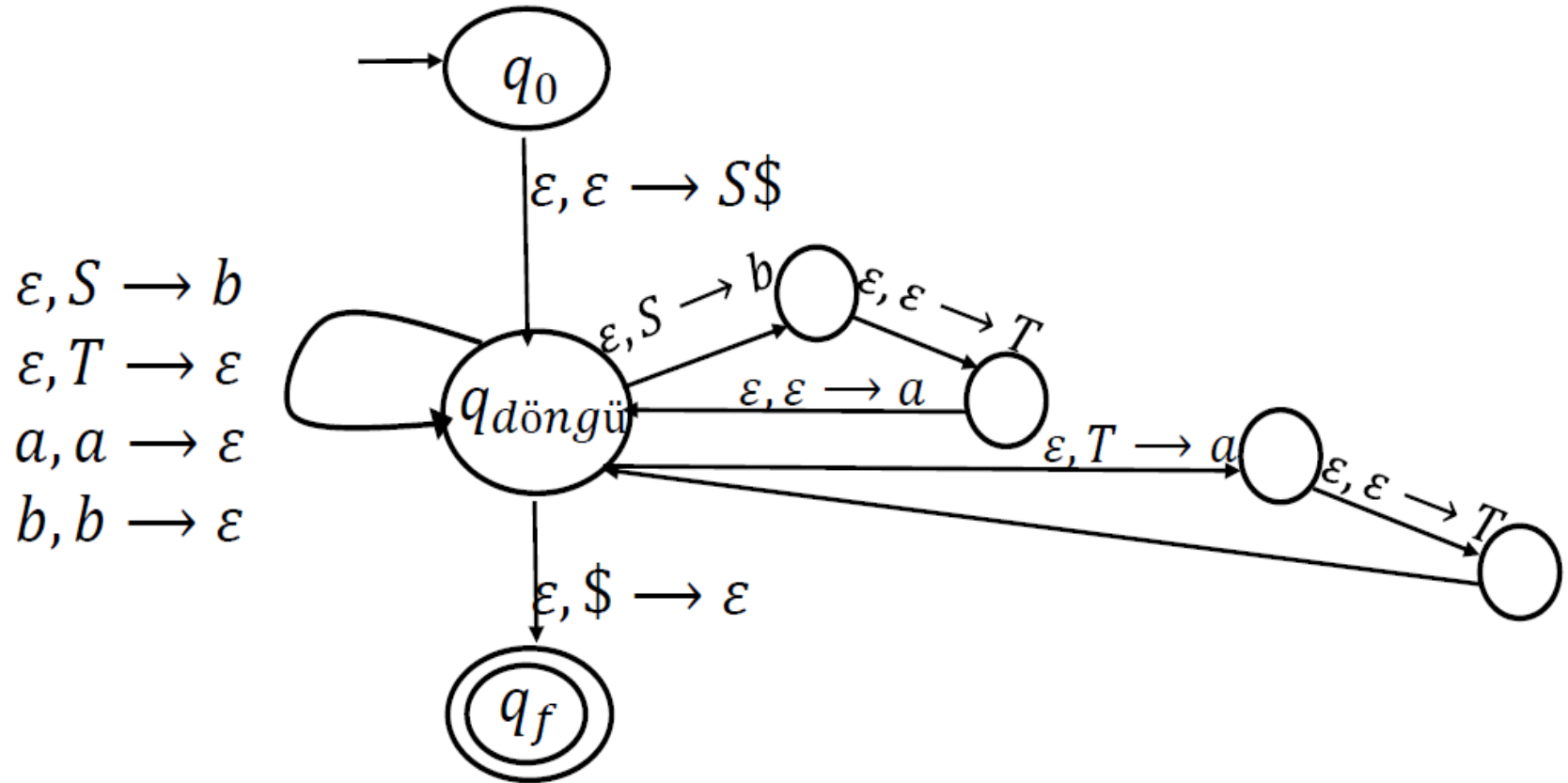
İçerikten Bağımsız Grammerlerden Pushdown Otomataya

ör. $G = (\{S, T\}, \{a, b\}, R, S)$ ve R kuralları

$$S \rightarrow aTb|b$$

$$T \rightarrow Ta|\varepsilon$$

İle verilen grammerden PDA elde edelim:



ör.

ör

$$S \rightarrow (E) \mid E + E \mid E * E \mid a$$

ile verilen grammerden PDA elde edelim:

