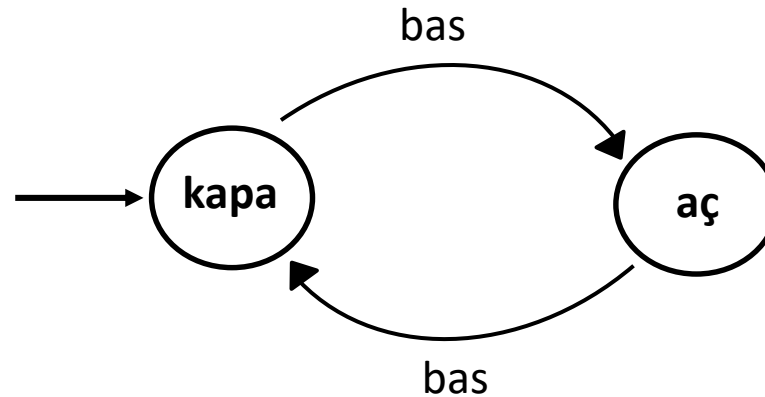


Otomata Teorisi

Fırat İsmailođlu, PhD

Hafta 2:

Sonlu Otomata (I.Bölüm)



Hafta 2

Plan

1. Bir Sonlu Otomata Orneđi
2. Sonlu Otomatanin Esasları
3. Sonlu Otomatanın Resmi Gösterimi
4. Nondeterministik Sonlu Otomata



I Bir Sonlu Otomasyon Örneği

Farz edelim ki bir makineye bağlı bir turnikemiz var ve bu makineye 25 kuruş atılınca turnike açılıyor.

Makine 5, 10 ve 25 kuruşluk madeni para kabul ediyor ve para üstü vermiyor.

Müşteri 5, 10 ve 25 kuruşluk paraları atarken her bir anda turnikenin açık yada kapalı oluşuna karar veren bir sonlu otomasyon tasarlıyalım.

Durumlar:

q_0 : Makinede para yok.

q_5 : Makinede 5 kuruş var.

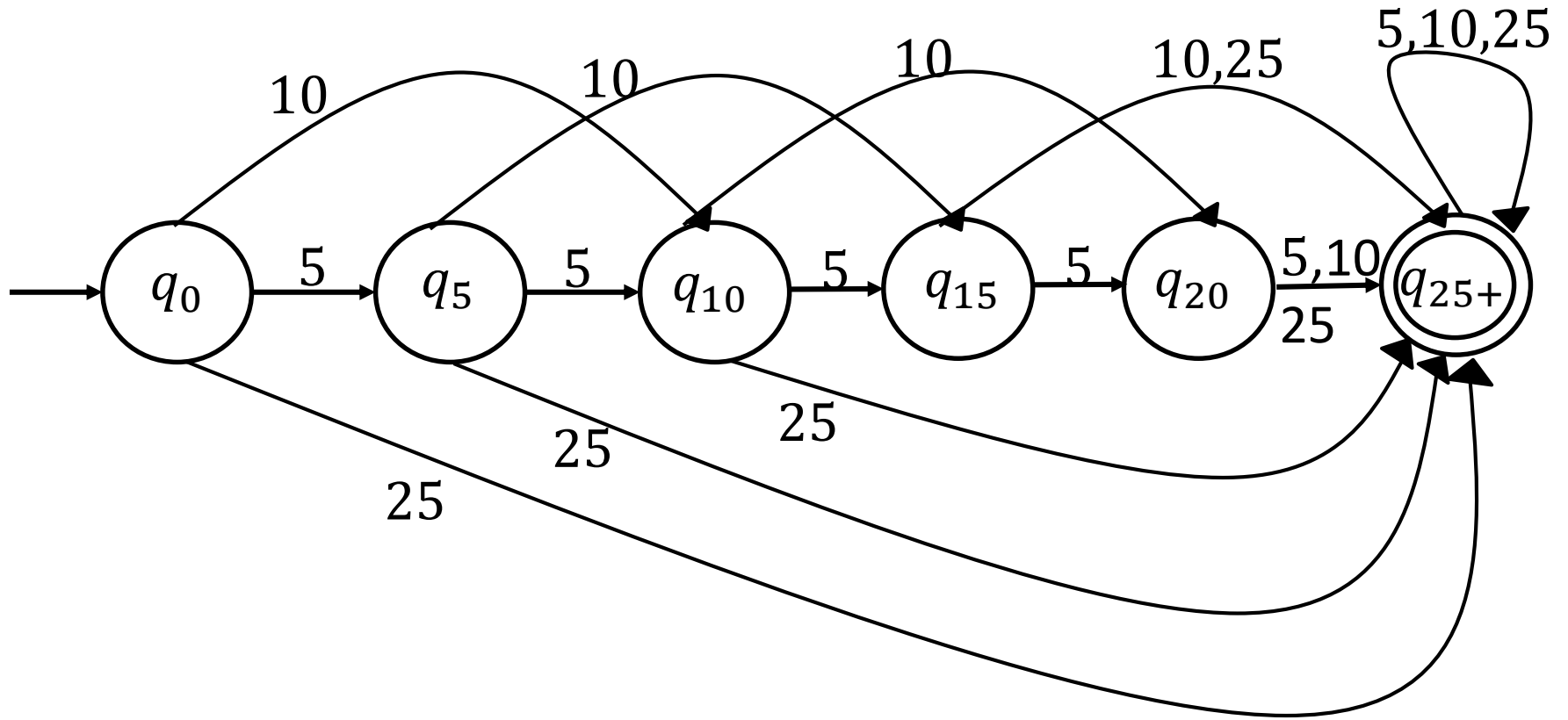
q_{10} : Makinede 10kuruş var.

q_{15} : Makinede 15 kuruş var.

q_{20} : Makinede 20 kuruş var.

q_{25+} : Makinede 25 kuruş yada daha fazlası var.





Burada makine (bilgisayar) yalnızca herhangi bir anda hangi durumda olduğunu hatırlamak zorunda. O yüzden yalnızca çok küçük bir hafızaya ($\log 6 \approx 3$ bit) ihtiyaç duyar.



Dil: Bir Σ alfabesinden elde edilebilecek tüm kelimelerin kumesi olan Σ^* un bir alt kumesine dil denir.

or. $\Sigma = \{0,1\}$ alfabeti için bu alfabeden üretilebilecek tüm kelimelerin kumesi:

$$\Sigma^* = \{\varepsilon, \mathbf{0}, \mathbf{1}, \mathbf{00}, \mathbf{11}, \mathbf{01}, \mathbf{10}, \mathbf{000}, \mathbf{111}, \mathbf{010}, \dots\}$$

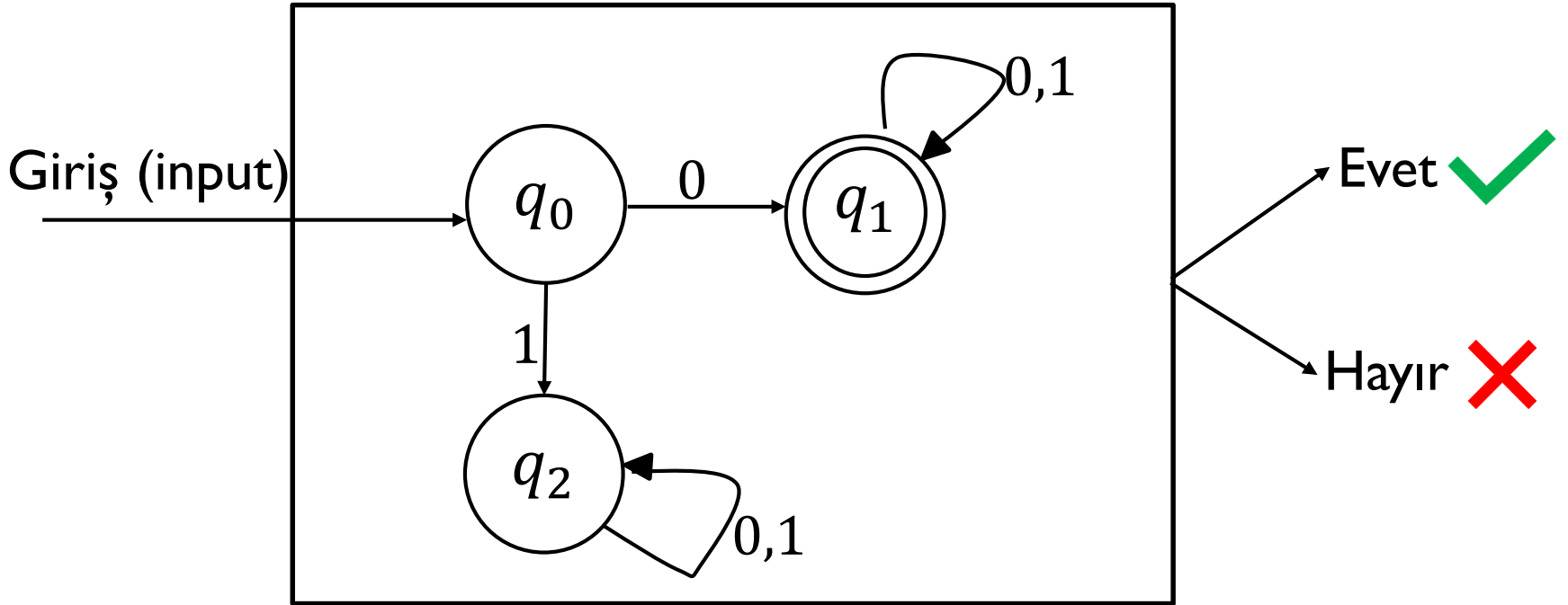
Buradan bir L dili oluşturulalım, öyleki bu L dili “0” ile başlayan kelimelerin dili olsun:

$$L = \{0, 00, 01, 000, 0101, \dots\}$$

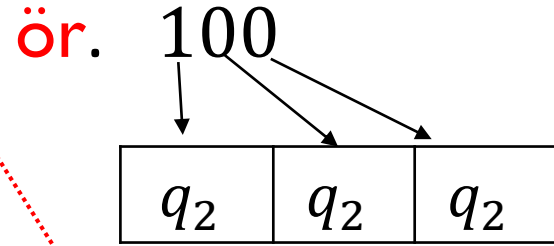
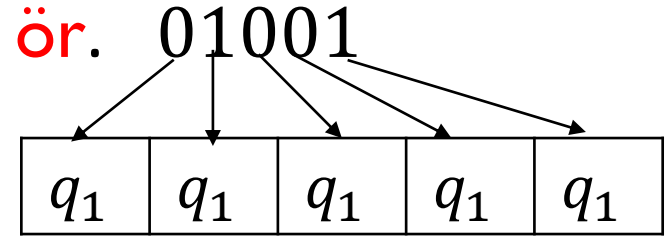
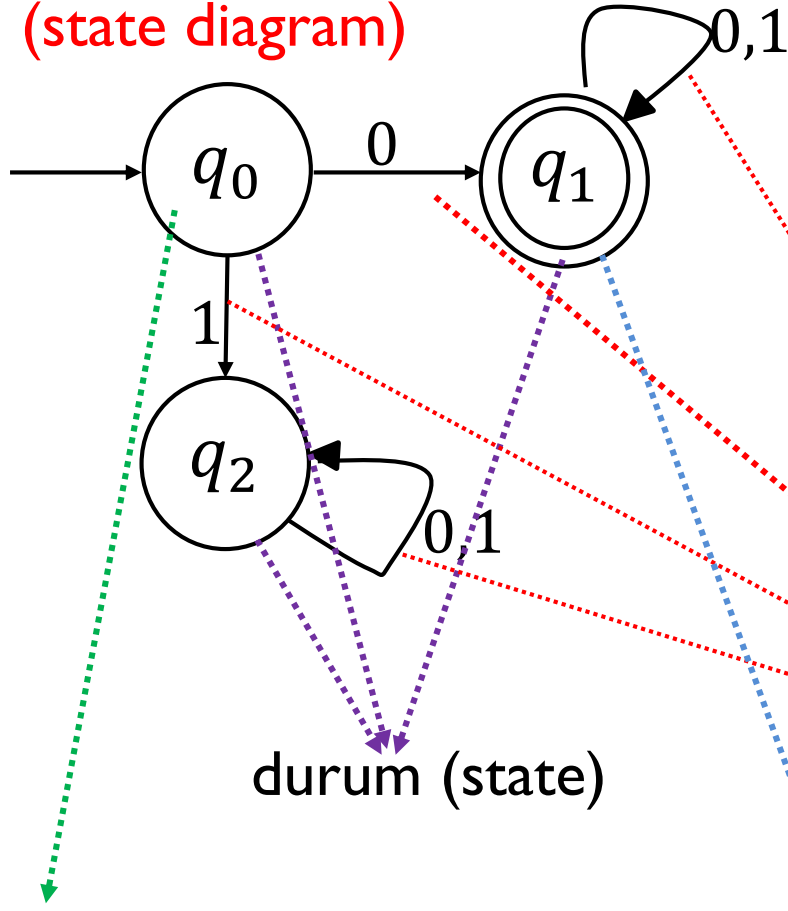
($L = \{s \in \Sigma^* \mid s'nin\ ilk\ harfi\ 0\}$ şeklinde ortak özellikler yöntemi ile de gösterebiliriz.)



İşte otomasyonda amacımız verilen bir dilin kelimelerini otomatik olarak algılayan-taniyan (recognize eden) bir soyut makine icat etmektir. İcat edilen makine verilen bir kelimeyi işler (process eder), ve sonuçta bu kelime dile ait mi diye karar verir (evet yada hayır der).



durum diagrami
(state diagram)



durum (state)

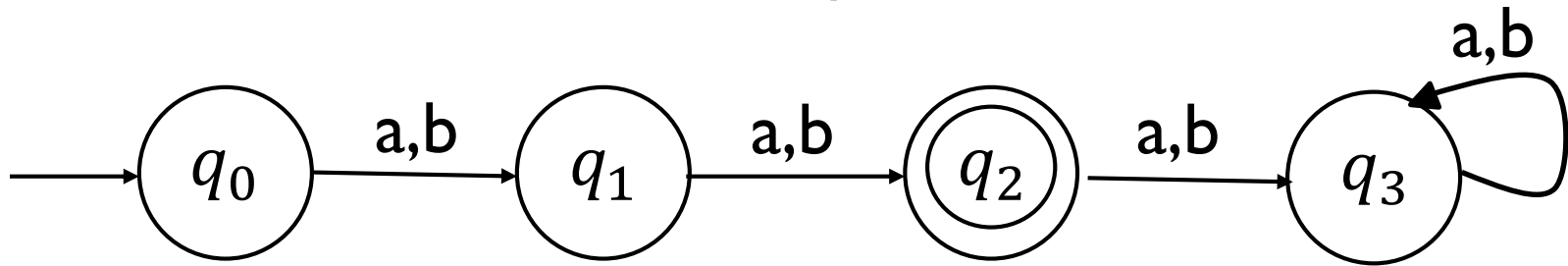
geciş (transition)

başlangıç durumu
(initial state)
(boş ok ile gösterilir)

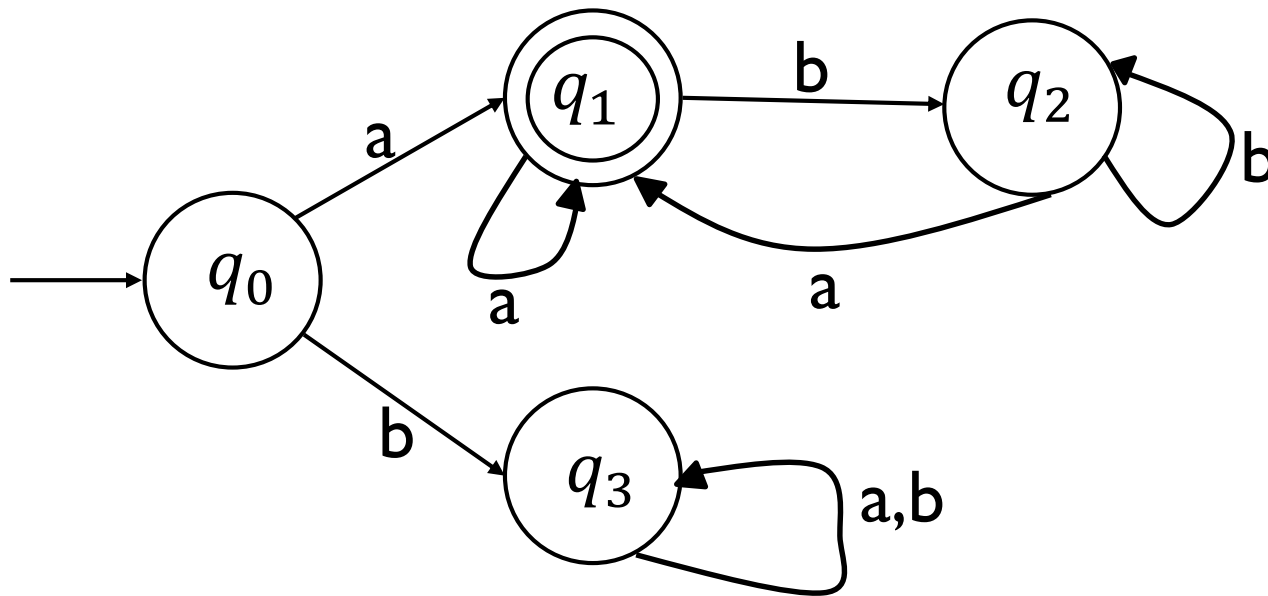
final (kabul) durumu
(final (acceptance) state)
(çift halka ile gösterilir)



ör. $\Sigma = \{a, b\}$ alfabeti kullanılarak üretilen kelimelerden 2 uzunluğundaki kelimelerin dilini tanıyan sonlu otomata:



ör. $\Sigma = \{a, b\}$ alfabeti kullanılarak üretilen kelimelerden a ile başlayıp a ile biten kelimelerin dilini tanıyan sonlu otomata:



Sonlu Otomatinin Formal Tanımı:

Bir sonlu otomata 5-li sıradır ve $(Q, \Sigma, \delta, q_0, F)$ ile gösterilir. Burada:

1. Q tüm durumları içeren sonlu bir kümedir,
2. Σ kullandığımız harfleri (inputları) içeren alfabledir,
3. $\delta: Q \times \Sigma \rightarrow Q$ geçiş fonksiyonudur (transition function),
4. $q_0 \in Q$ başlangıç durumudur,
5. $F \subseteq Q$ final durumları içeren kümedir.

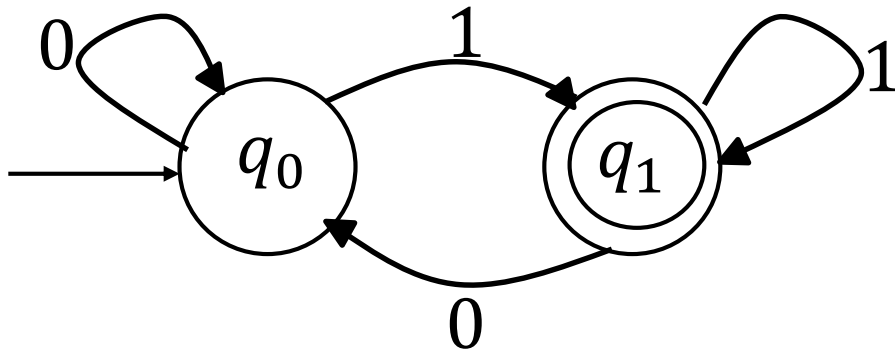
Not 1. δ (geciş fonksiyonu) bir fonksiyondur. Fonksiyon olma tanımı gereği $Q \times \Sigma$ tanım kümesindeki her elemanı (her sıralı ikiliyi) Q 'da bir elemana götürür. Yani her durum ve her harf ikilisine karşılık bir durum vardır.

Not 2. F bir kümedir. Yani eleman sayısı 1'den fazla olabilir. Yani bir otomatanın 1'den fazla final durumu (kabul durumu) olabilir.

Not 3. Sonlu otomata dememizin nedeni sonlu sayıda durum içermesi



ör.



$$Q = \{q_0, q_1\}$$

$$\Sigma = \{0,1\}$$

$$F = \{q_1\}$$

harfler

δ	harfler	
	0	1
durumlar	q_0	q_1
	q_1	q_1

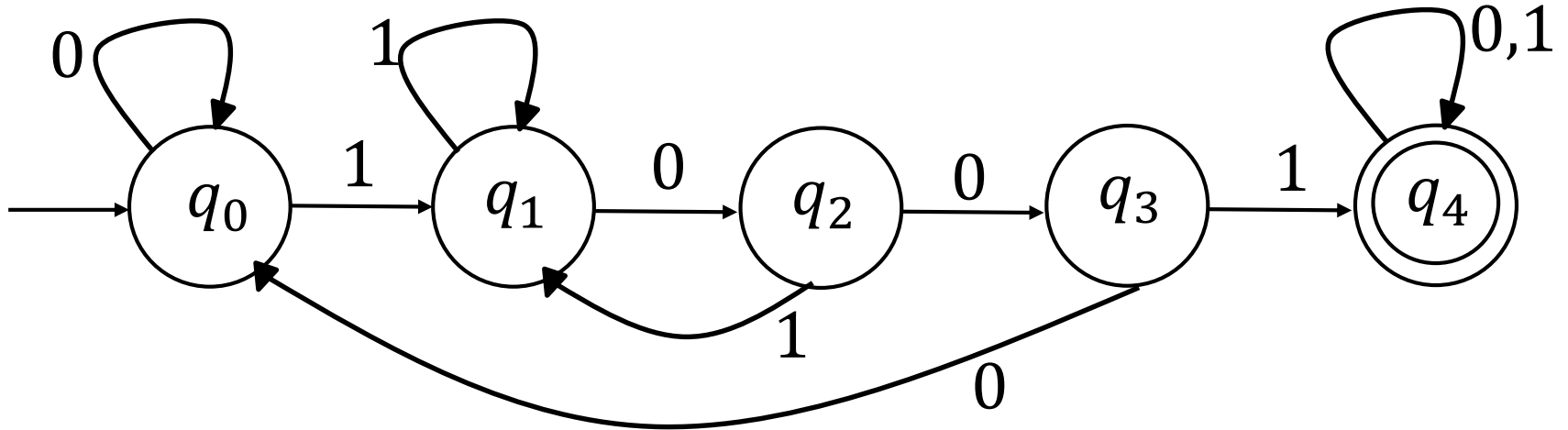
$\delta(q_0, 1) = q_1$

geçiş tablosu

$$M = \{\{q_0, q_1\}, \{0,1\}, \delta, q_0, \{q_1\}\}$$



ör. $\Sigma = \{0,1\}$ alfabeti kullanılarak üretilen kelimelerden, içinde '1001' altkelimesi olan kelimeleri tanıyan otomasyon.



$$Q = \{q_0, q_1, q_2, q_3, q_4\}$$

$$\Sigma = \{0,1\}$$

$$F = \{q_4\}$$

δ	0	1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_3	q_1
q_3	q_0	q_4
q_4	q_4	q_4



Bir kelimenin bir sonlu otomata tarafından kabul edilirligi

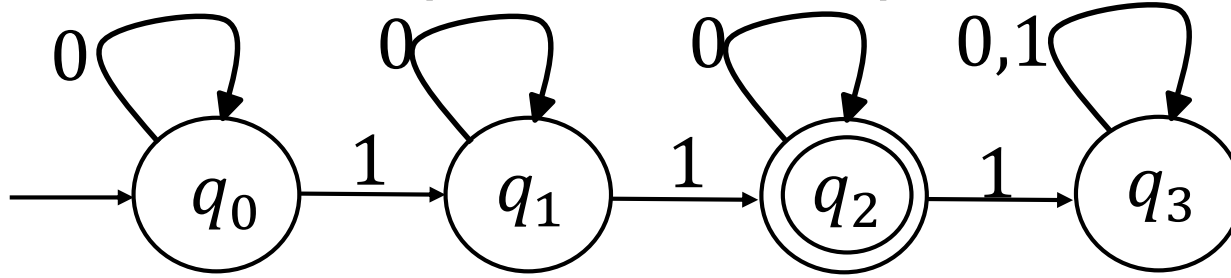
giriş: $M = (Q, \Sigma, \delta, q_0, F)$ makinesi, w kelimesi

çıkış: Evet yada Hayır

1. $q_{\text{şimdiki}} := q_0$
2. for $i = 1: |w|$ // w 'nun her bir harfi için
3. $q_{\text{yeni}} := \delta(q_{\text{şimdiki}}, w_i)$; //şimdiki durumdan w_i ile yeni duruma geç
4. $q_{\text{şimdiki}} := q_{\text{yeni}}$ // yeni durumu şimdiki durum yap (şimdiki durumu güncelle)
5. end for
6. if $q_{\text{şimdiki}} \in F$: // eğer son durum final durumlarından biriye
7. cout<<" Evet"; // Evet yazdır
8. else // değilse
9. cout<<"Hayır"; // Hayır yazdır
10. end if



ör. $\Sigma = \{0,1\}$ alfabeti kullanılarak üretilen kelimelerden, içinde 2 tane 1 olan kelimeleri tanıyan sonlu otomasyon:



Yukarıdaki otomasyon $w=1011$ kelimesini kabul eder mi?

1. $q_{\text{şimdiki}} := q_0$

2. for'a başla:

$$\begin{array}{l}
 i = 1 \left\{ \begin{array}{l} q_{\text{yeni}} := \delta(q_0, 1) = q_1 \\ q_{\text{şimdiki}} = q_1 \end{array} \right. \quad \begin{array}{l} \swarrow w_1 \\ \swarrow w_2 \end{array} \\
 i = 2 \left\{ \begin{array}{l} q_{\text{yeni}} := \delta(q_1, 0) = q_1 \\ q_{\text{şimdiki}} = q_1 \end{array} \right. \\
 i = 3 \left\{ \begin{array}{l} q_{\text{yeni}} := \delta(q_1, 1) = q_2 \\ q_{\text{şimdiki}} = q_2 \end{array} \right. \quad \begin{array}{l} \swarrow w_3 \\ \swarrow w_4 \end{array} \\
 i = 4 \left\{ \begin{array}{l} q_{\text{yeni}} := \delta(q_2, 1) = q_3 \\ q_{\text{şimdiki}} = q_3 \end{array} \right.
 \end{array}$$

3. for'u bitir.

4. $q_{\text{şimdiki}} = q_3 \notin F$ olduğundan cevap Hayır.

Nondeterministik (Belirsiz) Sonlu Otomata (NSO)

Şuana kadar gördüğümüz sonlu otomata "Deterministik Sonlu Otomata (DSO) " idi. (Deterministic Finite Automata (DFA)) (Deterministik rasteyele olmayan, kararlı, belirli şekilde Türkçe'ye çevriliyor.)

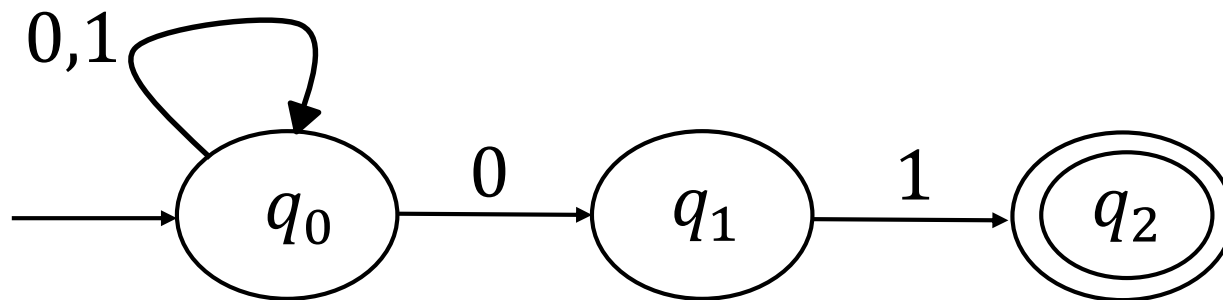
DSO'nun ana karakteristiği, belirli bir anda makinenin yalnızca bir durumda bulunabilir olmasıdır (iki durum aynı anda aktif olamaz).

NSO, DSO kavramının genelleştirilmesidir: $DSO \subseteq NSO$, yani her DSO bir NSO'dur. Ayrıca her NSO, bir DSO ya dönüştürülebilir.

Bir NSO da makine aynı anda birden fazla durumda olabilir. Bu NSO'nun ana karakteristiği dir. Bu bize daha karmaşık dilleri tanıyabilen makineler üretebilmeyi sağlar.



ör. $\Sigma = \{0,1\}$ alfabeti kullanılarak üretilen kelimelerden sonu "01" ile biten kelimeleri tanıyan sonlu otomata:



soru: bu otomatada dikkatimizi çeken şeyler neler?

1. q_0 ' da 2 tane 0 oku var. Yani q_0 'da iken 0 gelirse hem q_0 'a, hem q_1 'e geçeriz.
2. q_1 'de 0 oku yok. Yani q_1 'de 0 gelirse makine (otomata) çıkmaza girer (ölür).
3. q_2 'de ne 0 oku nede 1 oku var. Yani makine q_2 'de sonlanmaz harf almaya (0,1) almaya devam ederse, makine çıkmaza girer.

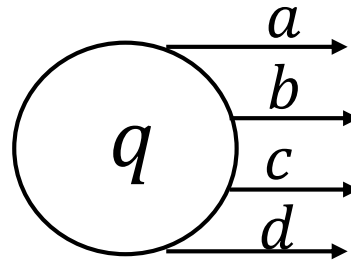


Hatırla: Daha önce gördüğümüz deterministik sonlu otomatada (DSO) geçiş fonksiyonu:

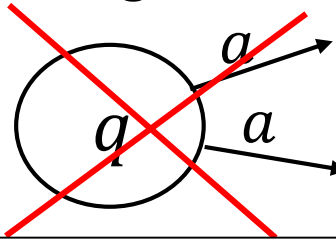
$$\delta: Q \times \Sigma \rightarrow Q$$

1. Burada δ bir fonksiyon olduğundan her bir (durum, harf) ikilisi için fonksiyon tanımlıdır. Yani her bir durum ($q \in Q$) da bütün harfler ($\forall s \in \Sigma$) için bir ok yani bir geçiş vardır:

ör. $\Sigma = \{a, b, c, d\}$ için



2. Yine, δ bir fonksiyon olduğundan aldığı her bir (durum, harf) ikilisini bir ve yalnız bir duruma götürür.

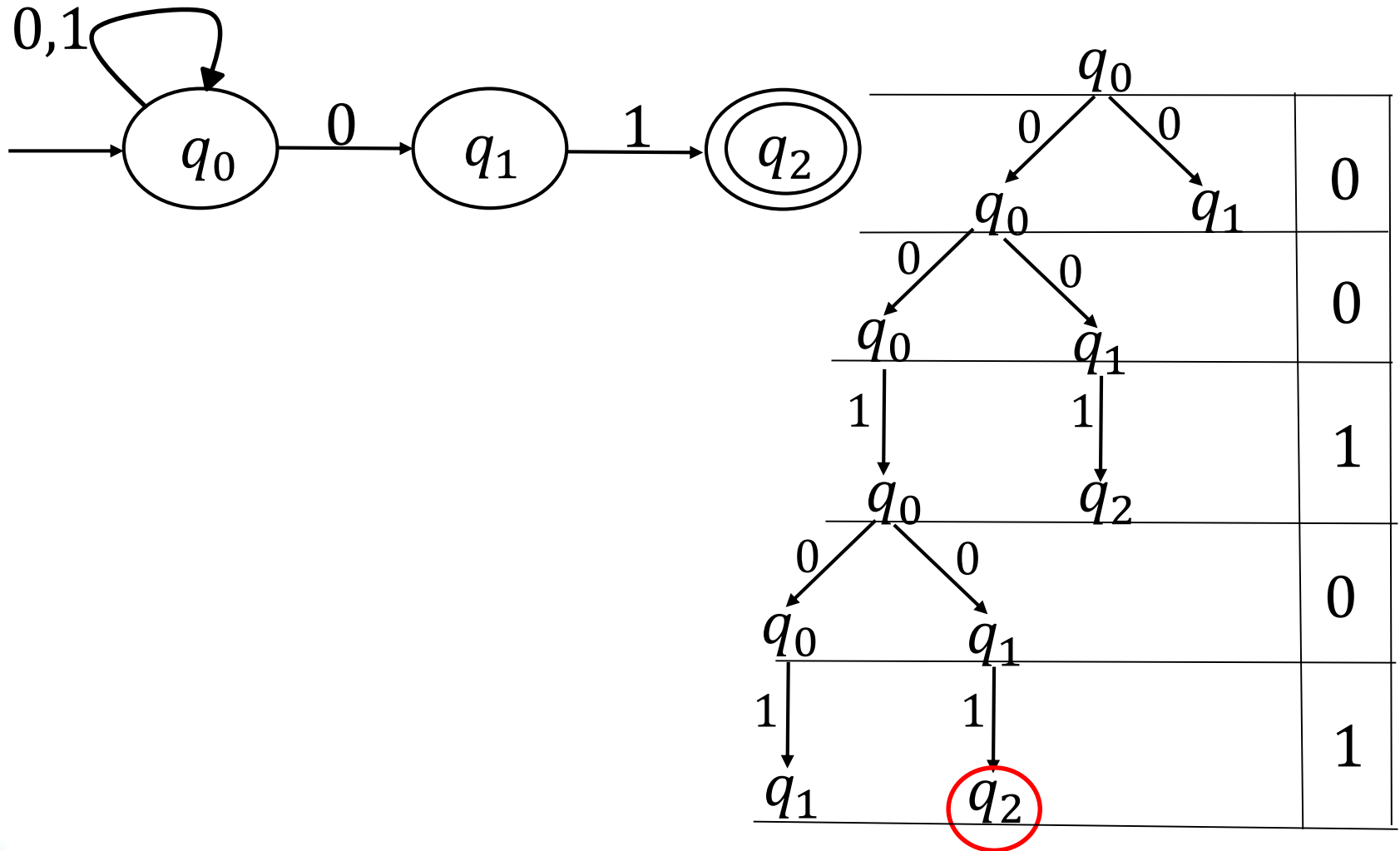


!! NSO'da bu yukarıdaki durumların ikisinde geçerli değildir.

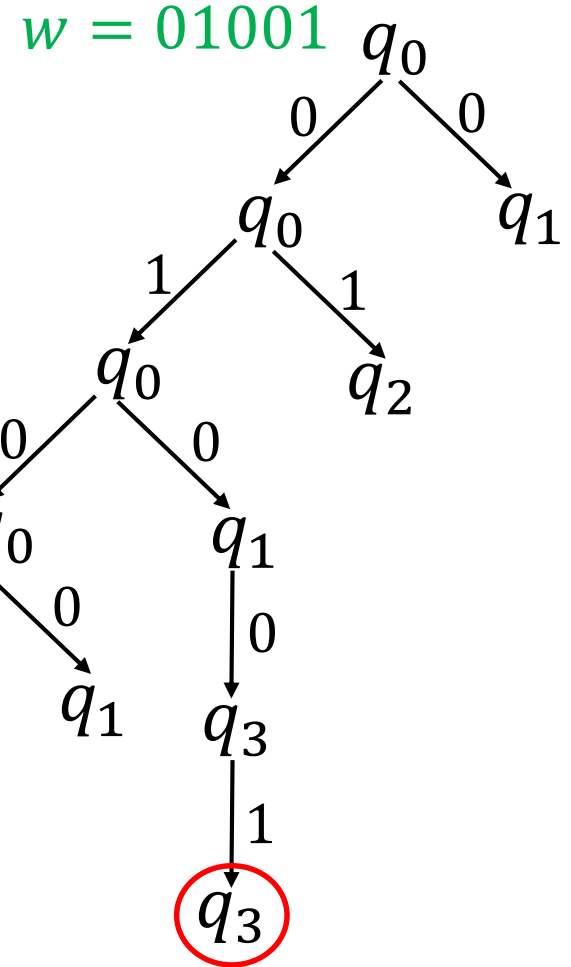
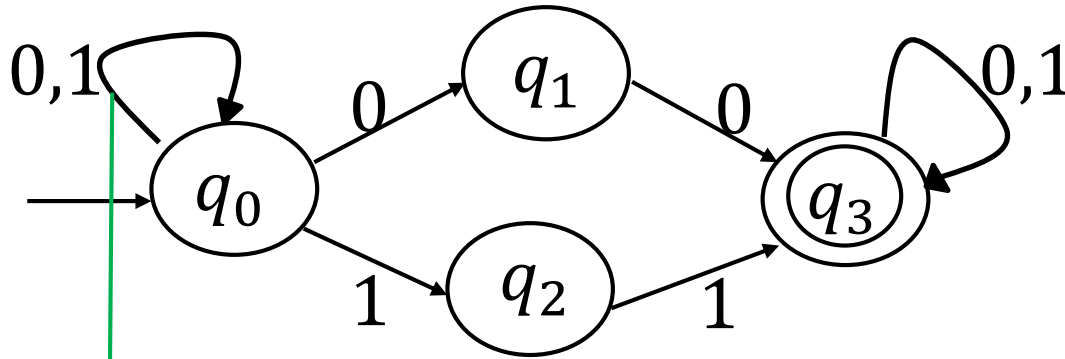


Bir NSO'nun İşleyişi

$w = 00101$ kelimesi aşağıdaki NSO şu şekilde işlenir:



ör. $\Sigma = \{0,1\}$ alfabeti kullanılarak üretilen kelimelerden içinde "00" yada "11" alt kelimeleri geçen kelimeleri tanıyan nondeterministik sonlu otomata:



Eğer q_0 'ın bu oku olmazsa, makine yalnızca 00 yada 11 ile başlayan kelimeleri kabul eder!

Bu işlemin sonunda makine q_0, q_2 ve q_3 durumlarında duruyor.

