

CENG 306 Biçimsel Diller ve Otomatlar

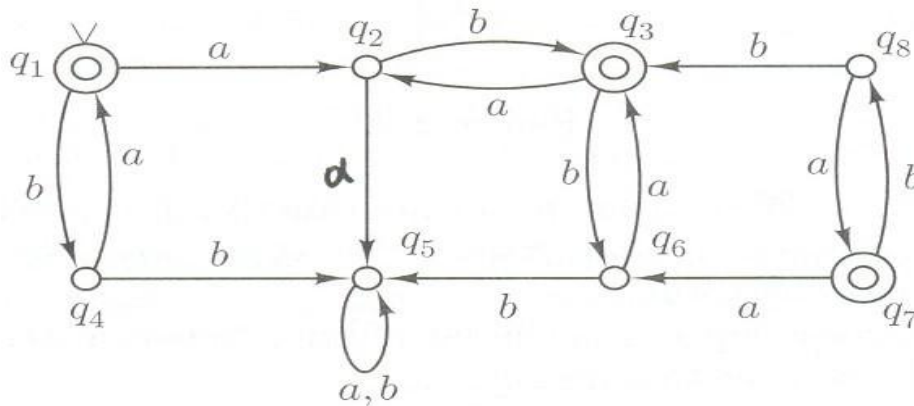
Formal Languages and Automata

Konular

- Durum İndirgeme - State Minimization
- Myhill-Nerode Theorem
- Table-Filling Algorithm

State Minimization

- Bir M otomatı için birçok durumu gözardı etmenin kolay bir yolu olabilir.
- Aşağıdaki otomat $L=(ab \cup ba)^*$ dilini tanıır.



- Burada q_7 ve q_8 durumları unreachable (erişilemez) durumdur.
- Unreachable durumların tamamı otomat'tan doğrudan çıkarılabilir.
- NFA'nın DFA eşitini bulurken aynı optimizasyon yapılmaktadır.

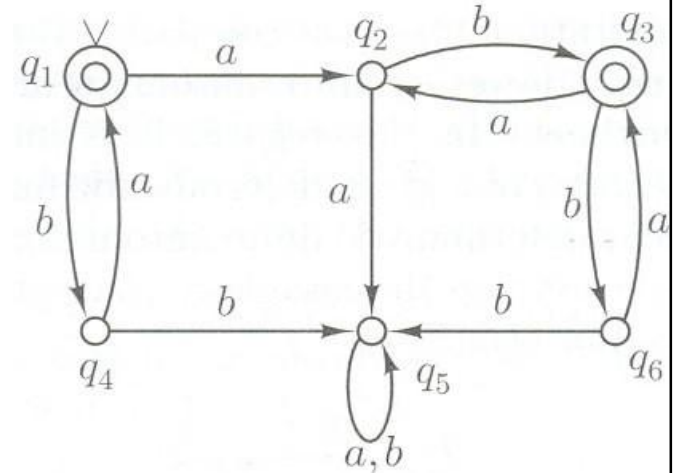
State Minimization

- Unreachable durumları bulan algoritma aşağıdaki gibidir;

$R := \{s\};$

while there is a state $p \in R$ and $a \in L$ and $\partial(p, a)$ such that $\partial(p, a) \notin R$
do add $\partial(p, a)$ to R

- Bu algoritmayla silinen durumlardan sonra da otomat hala gereksiz durumlara sahip olabilir.
- Burada q_4 ve q_6 durumları denk'tir (**equivalent**).
- Bu yüzden bir durum olarak birleştirilebilir.
- q_4 ve q_6 denk durumlar aynı string için otomatı sonuç durumuna götürür.
- Denk durumlar aynı string için otomatı sonuç olmayan farklı diğer durumlara da götürebilir.



State Minimization

Tanım: L diline göre iki string'in denkliği

$L \subseteq L^*$ ve $x, y \in L^*$ olsun. Eğer $z \in L^*$ ve $xz \in L$ iken $yz \in L$ olursa x ve y , **L diline göre denk** olarak adlandırılır, $x \approx_L y$ şeklinde gösterilir.

- x ve y string'lerinin ikisi de L diline ait olabilir veya olmayabilir.
- Sonlarına eklenen herhangi bir string x ve y 'i L diline ait yapabilir veya yapmayabilir.

State Minimization

Örnek: x bir string ve L bir dil ise, $[x]$ bu L diline göre x 'in sahip olduğu denk sınıfı gösterebilir.

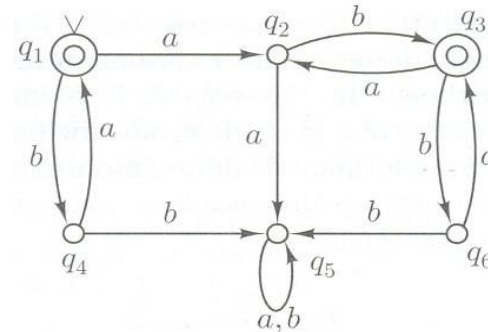
Aşağıdaki otomat $L = (ab \cup ba)^*$ dilini tanıtır ve \approx_L için 4 denk sınıf vardır;

1. $[e] = L$

2. $[a] = La$

3. $[b] = Lb$

4. $[aa] = L(aa \cup bb)^*$



- 1. **durumda** herhangi bir $x \in L$ için, $x = e$ bile olsa $xz \in L$ yapan her z string'ide L diline aittir.
- 2. **durumda** herhangi bir $x \in La$ string'i $xz \in L$ olabilmesi için z string'inin **bL** şeklinde olması gereklidir.
- 3. **durumda** z string'inin **bL** şeklinde olması gereklidir.
- 4. **durumda** hiçbir z string'i $L(aa \cup bb)$ prefix'i için bu string'i dile ait **yapamaz**.
- 1. **durumdaki kümeye ait tüm stringler aynı durumlara, ve diğer 2., 3., veya 4. durumlardaki string kümeleri de aynı durumlara götürür.**

State Minimization

Tanım: M otomatına göre iki string'in denkliği

$M = (K, L, \delta, s, F)$ bir deterministic automata olsun.

Eger iki string $x, y \in \Sigma^*$ M otomatını s başlangıç durumundan aynı duruma götürüyorsa **M otomatına göre denktir** ve $x \sim_M y$ şeklinde gösterilir.



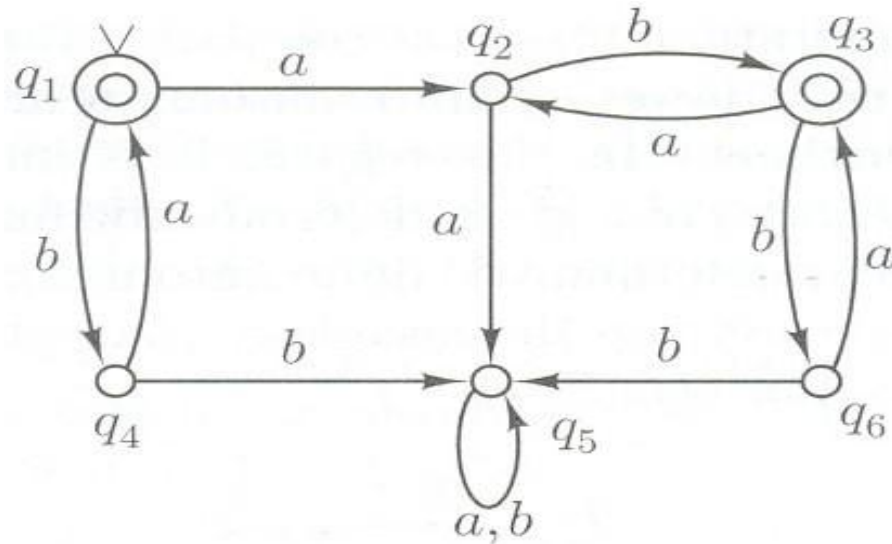
$(s, x) \vdash^*(q, e)$ ve $(s, y) \vdash^*(q, e)$ ise $x \sim_M y$ olur.

- Herhangi bir q durumu için denk sınıf E_q şeklinde gösterilir.
- Bu durumların s başlangıç durumundan erişilebilir olması zorunludur.

State Minimization

Örnek: Aşağıdaki otomat $L = (ab \cup ba)^*$ dilini tanır ve \sim_M için 6 **denk sınıf** vardır;

- 1. $E_{q_1} = (ba)^*$
- 2. $E_{q_2} = La \cup a$
- 3. $E_{q_3} = abL$
- 4. $E_{q_4} = b(ab)^*$
- 5. $E_{q_5} = L(bb \cup aa) \Sigma^*$
- 6. $E_{q_6} = abLb$



Theorem: Deterministic $M = (K, L, \delta, s, F)$ otomatında herhangi $x, y \in \Sigma^*$ string'leri için $x \sim_M y$ ise $x \approx_{L(M)} y$ olur.

Myhill-Nerode Theorem

Teorem: $L \subseteq \Sigma^*$ regular dil olsun. **L dilini tanıyan ve \approx_L içindeki denk sınıfların sayısına tam eşit sayıda duruma sahip olan bir deterministic automata vardır.**

ispat: $x \in \Sigma^*$ string'i için \approx_L ilişkisi içinde denk sınıflar $[x]$ şeklinde gösterilir. Bu dil için

$M = (K, \Sigma, \delta, s, F)$ otomatı aşağıdaki gibi oluşturulabilir;

$K = \{ [x] : x \in \Sigma^* \}$, \approx_L altında denk sınıflar kümesi

$s = [e]$, \approx_L altında e için denk sınıflar kümesi

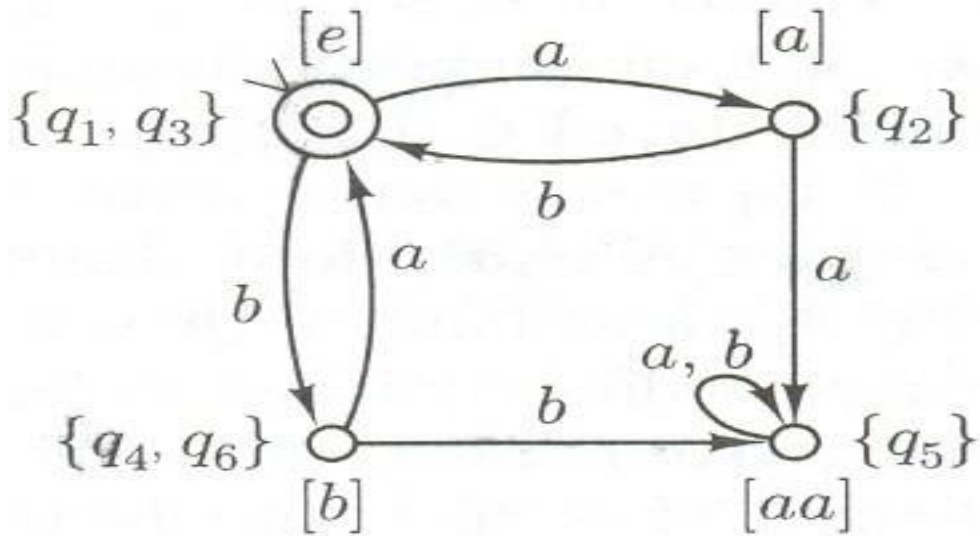
$F = \{ [x] : x \in L \}$,

Son olarak herhangi bir $[x] \in K$ ve herhangi bir $a \in \Sigma$ için,

$\delta([x], a) = [xa]$ geçişleri tanımlanır.

Myhill-Nerode Theorem

Örnek: $L = (ab \cup ba)^*$ dilini tanıyan 6 duruma sahip olan deterministic otomat 4 durumla gösterilebilir.

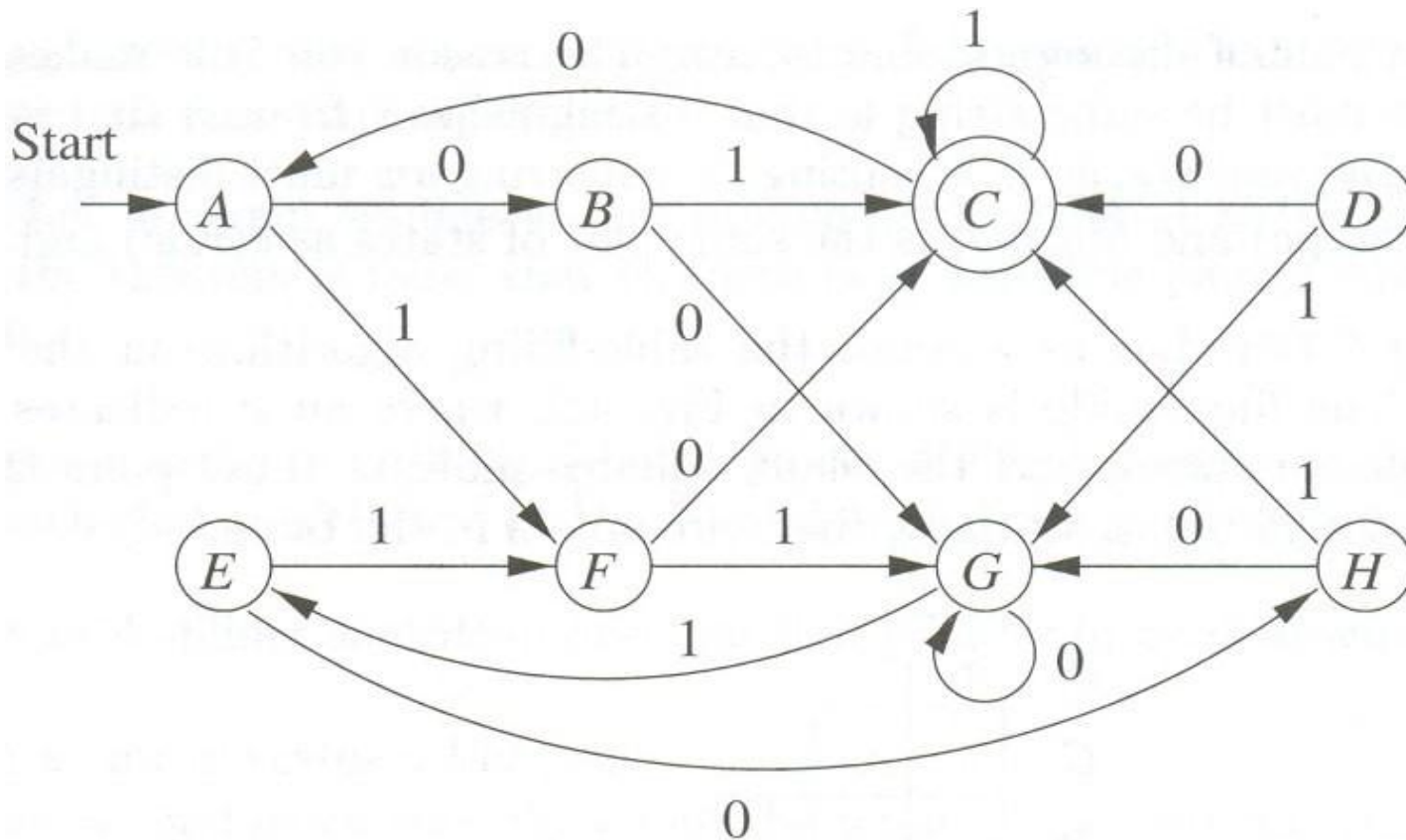


Corollary: L bir regular dildir, eger \approx_L sonlu sayıda denk sınıfa sahipse.

ispat : $L = L(M)$ regular ise, en az \approx_L deki denk sınıfların sayısı kadar duruma sahip bir M deterministic sonlu otomat vardır.

State Minimization

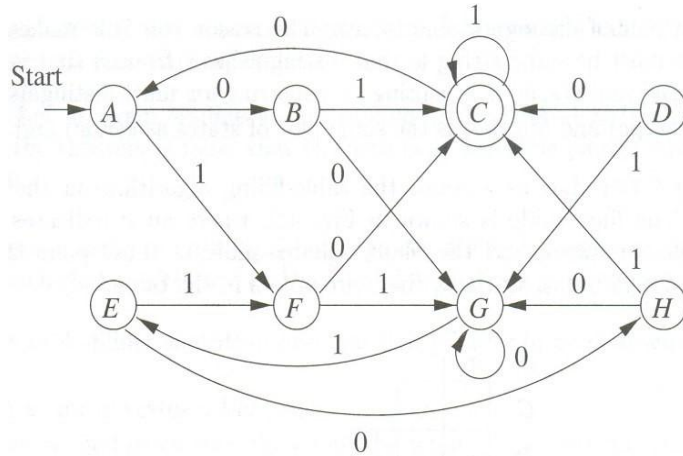
Örnek: Aşağıdaki otomat'ın en az duruma sahip eşitini bulalım. Denk durum var mıdır ?



State Minimization

Örnek: Aşağıdaki otomat'ın en az duruma sahip eşitini bulalım.

denk durum var mıdır ? C ve G denk degildir!



A ve G denk midir?

- e-string için ikisi denktir.

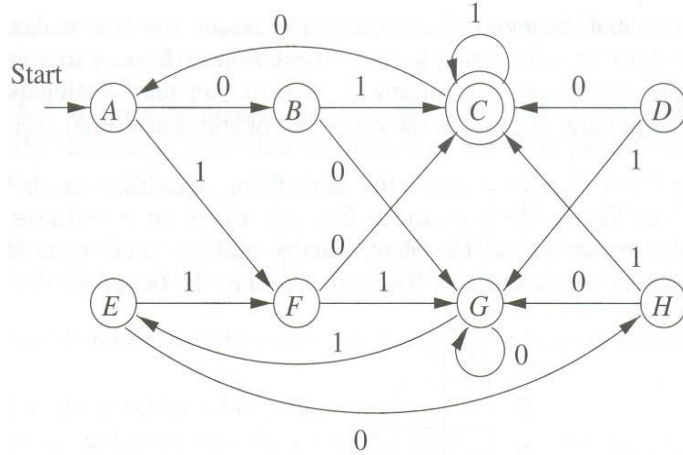
Çünkü ikisi de final state degildir.

- 0 için ikisi B ve G ye gider. İkisi de final state değildir ve denktir.
- 1 için F ve E ye giderler. İkisi de final state değildir ve denktir.

State Minimization

Örnek: Aşağıdaki otomat'ın en az duruma sahip eşitini bulalım.

denk durum var mıdır ? C ve G denk değildir!



A ve G denk midir?

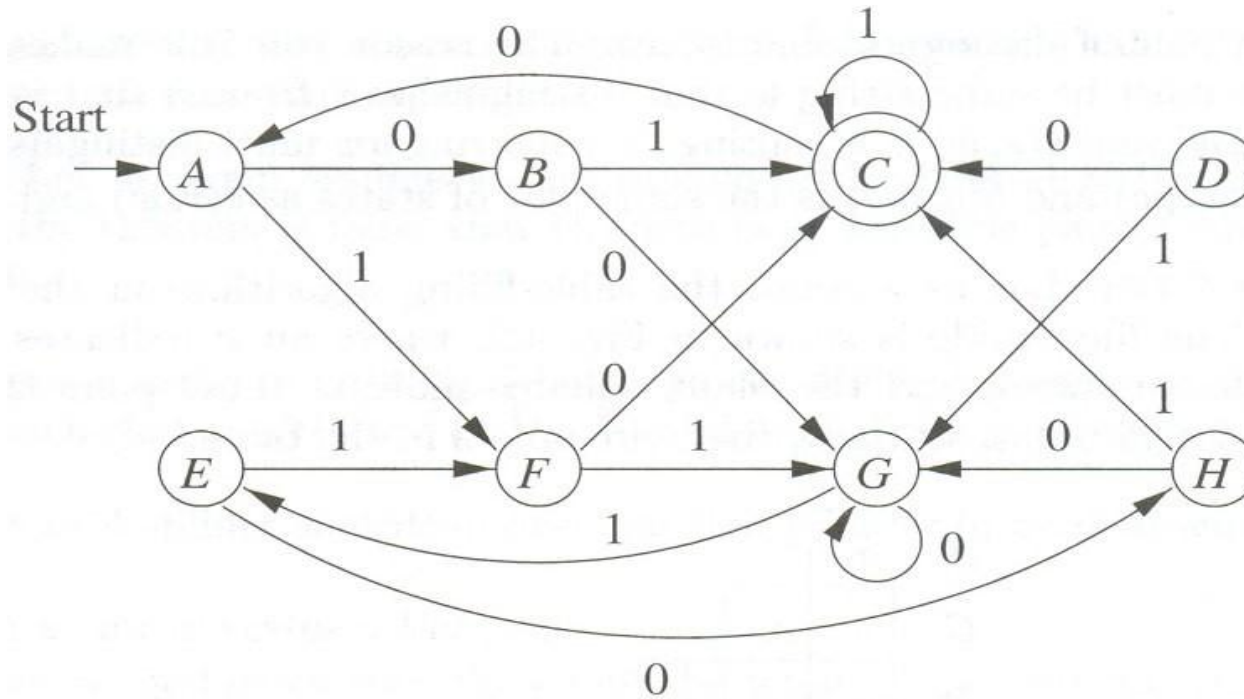
- e-string için ikisi denktir.

Çünkü ikisi de final state değildir.

- 0 için ikisi B ve G ye gider. İkisi de final state değildir ve denktir.
- 1 için F ve E ye giderler. İkisi de final state değildir ve denktir.
- 01 için sırasıyla C ve E ye giderler. C final state E değildir. Böylece **01 için denk değildirler.**
- Herhangi bir string için seçilen iki durumdan birisi final state'e giderken diğeri gitmiyorsa denk olmadıkları ispat edilmiş olur.

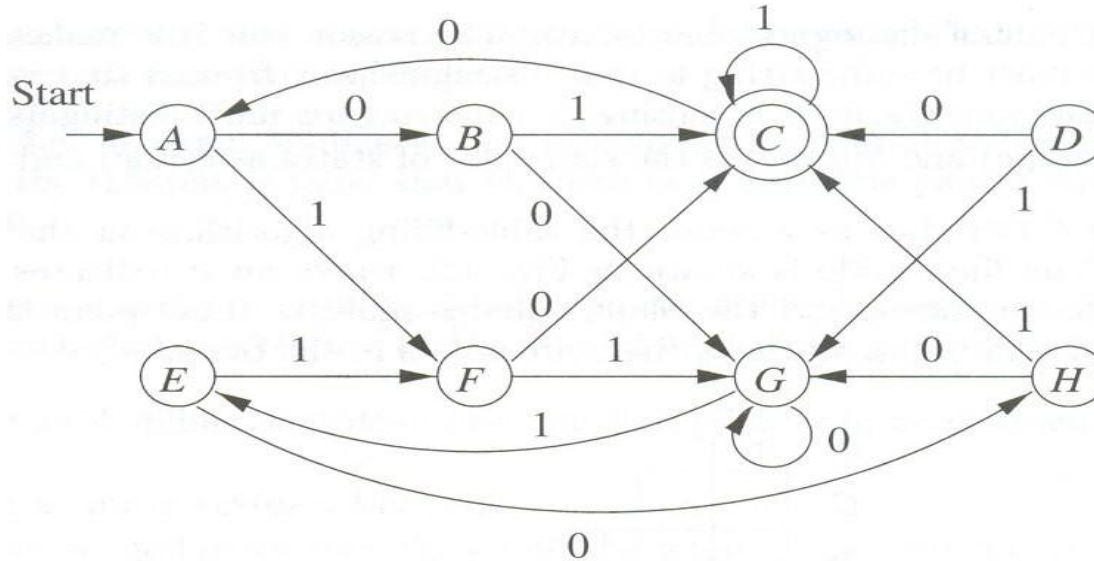
State Minimization

Örnek: Aşağıdaki otomat'ın en az duruma sahip eşitini bulalım.



State Minimization

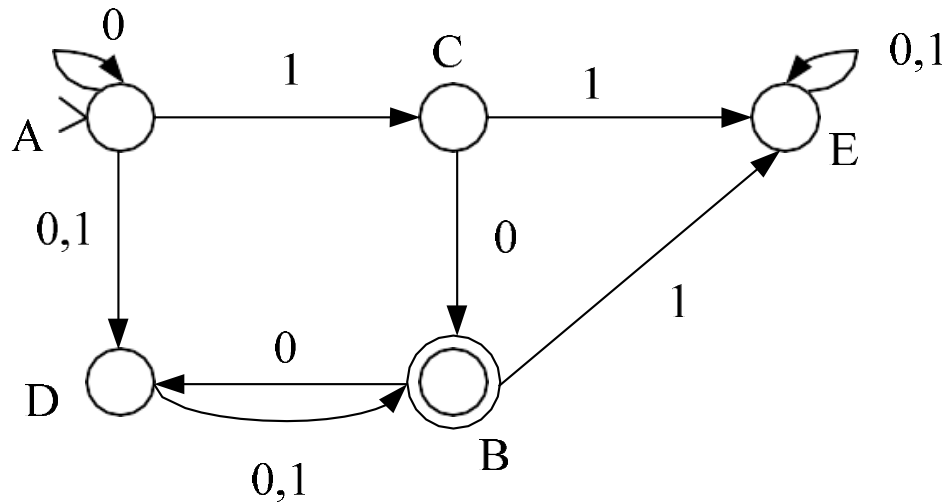
Örnek: Aşağıdaki otomat'ın en az duruma sahip eşitini bulalım.



- A ve E durumlarına bakalım. İkisi de final state olmadığı için e-string için denktirler.
- 1 için ikisi de F'ye gider. 1'le başlayan tüm stringler için denktirler.
- 0 için B ve H'ye giderler. İkisi de final state olmadığı için denktir.
- 01 için ikiside C'ye 00 için ikisi de G'ye gider. $\delta(A, \Sigma^*) = \delta(E, \Sigma^*)$

State Minimization

- Denk durumların bulunması için tüm durumları tek tek test etmek gerekir. Bu zor ve zaman alıcıdır. Hata yapma olasılığı yüksektir.
- Bunu sistematik yapmak için **Tablo Doldurma Algoritması** kullanılır.
- Durumlar kendi kendisini içermeyecek şekilde bir **tablo** hazırlanır.
- Başlangıçta final state'ler ile diğerleri denk değil olarak işaretlenir.



B	x			
C	x	x		
D	x	x	x	
E	x	x	x	x
	A	B	C	D

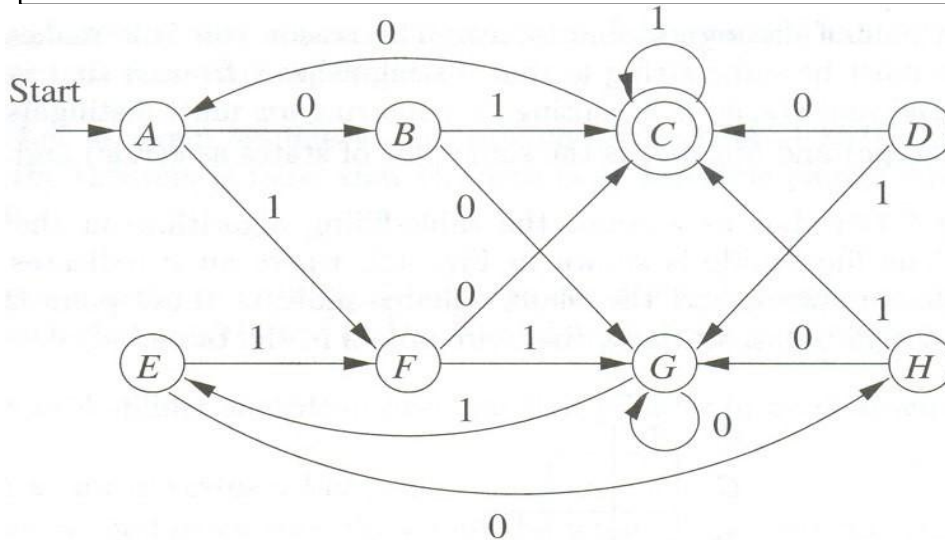
State Minimization

Örnek:

- *C final state'tir. C ile diğer durumlar arasında x konur.*
- *Diğer durum çiftlerinde 0 ve/veya 1 girişleri için final state ve diğer durumlara gidiliyorsa x ile işaretlenir.*

Örn: E ve F 0 için H ve C'ye gider. C final state ve H final state değil E ve F çifti işaretlenir.

- *Örneğin A ve G için 1 girişinde F ve E'ye gider. E-F çifti işaretli olduğu (denk olmadığı) için A-G işaretlenir.*



B							
C							
D							
E							
F							
G							
H							
	A	B	C	D	E	F	G

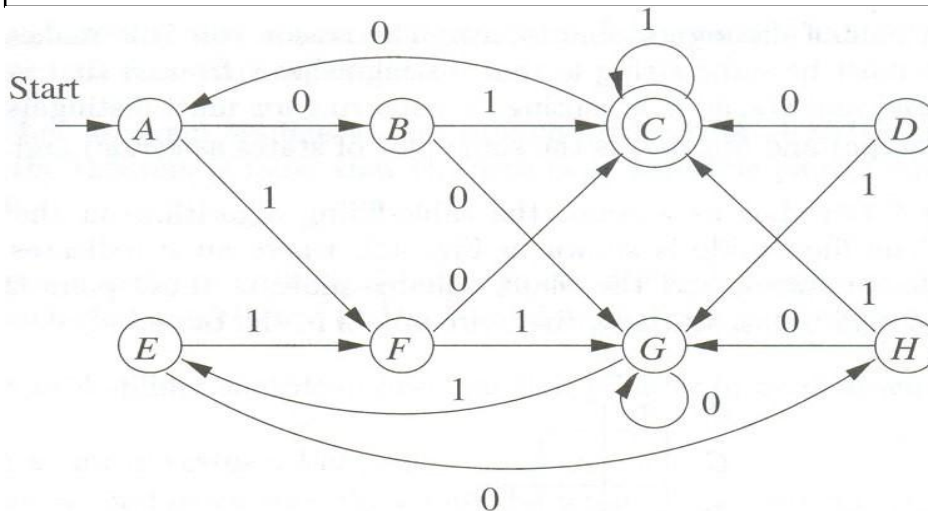
State Minimization

Örnek:

- *C final state'tir. C ile diğer durumlar arasında x konur.*
- *Diğer durum çiftlerinde 0 ve/veya 1 girişleri için final state ve diğer durumlara gidiliyorsa x ile işaretlenir.*

Örn: E ve F 0 için H ve C'ye gider. C final state ve H final state değil E ve F çifti işaretlenir.

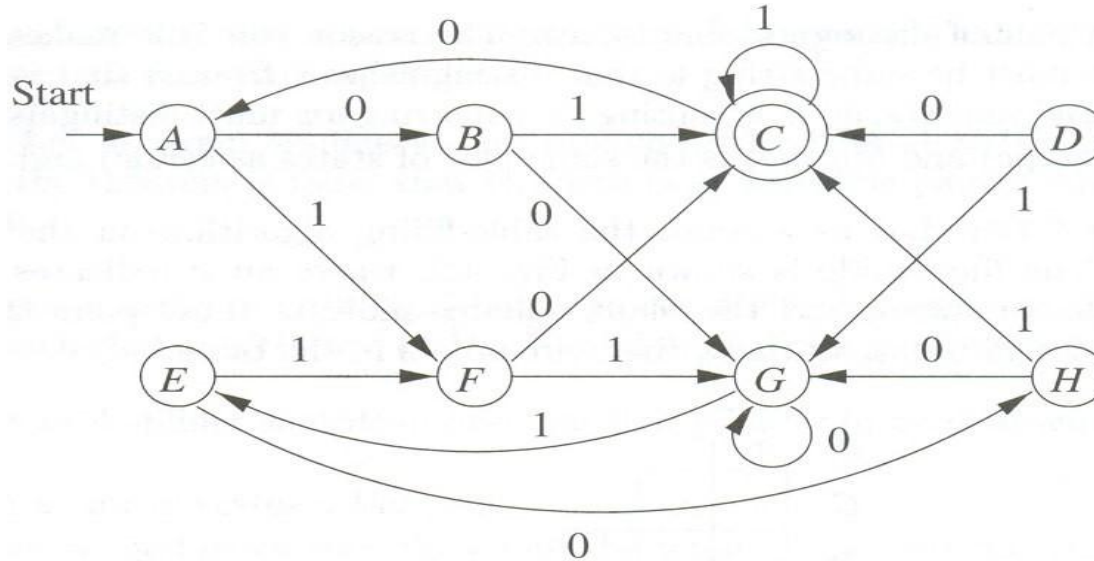
- *Örneğin A ve G için 1 girişinde F ve E'ye gider. E-F çifti işaretli olduğu (denk olmadığı) için A-G işaretlenir.*



B	x						
C	x	x					
D	x	x	x				
E		x	x	x			
F	x	x	x		x		
G	x	x	x	x	x	x	
H	x		x	x	x	x	x
	A	B	C	D	E	F	G

State Minimization

Örnek: Aşağıdaki otomat'ın en az duruma sahip eşitini bulalım.

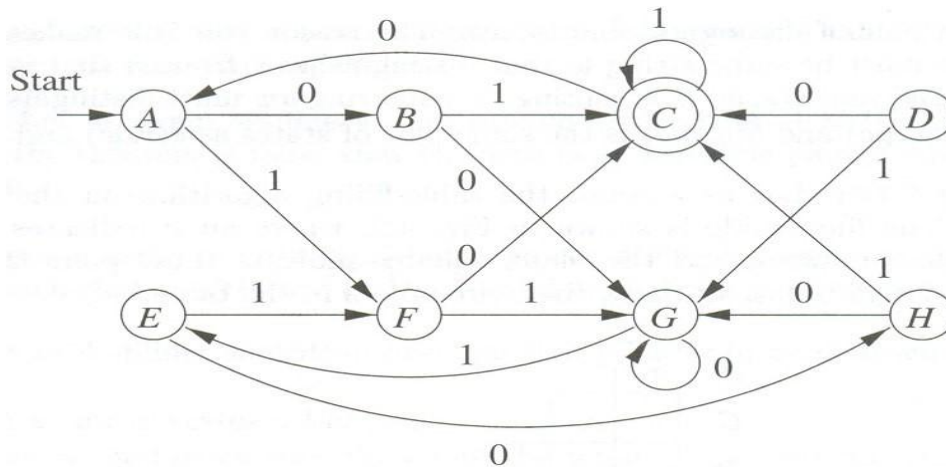


- *A ve E durumlarına bakalım. İkisi de final state olmadığı için e-string için denktirler.*
- *1 için ikisi de F'ye gider. 1'le başlayan tüm stringler için denktirler.*
- *0 için B ve H'ye giderler. İkisi de final state olmadığı için denktir.*
- *01 için ikiside C'ye 00 için ikisi de G'ye gider. $\delta(A, \Sigma^*) = \delta(E, \Sigma^*)$*

State Minimization

Örnek: (devam)

- *A-E, B-H ve D-F durumları denk durumlardır ve birleştirilebilir.*

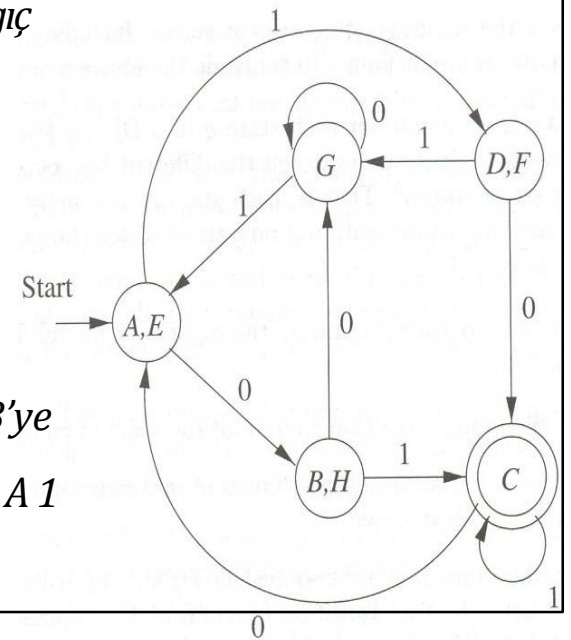


B	x						
C	x	x					
D	x	x	x				
E		x	x	x			
F	x	x	x		x		
G	x	x	x	x	x	x	
H	x		x	x	x	x	x
	A	B	C	D	E	F	G

State Minimization

Örnek: (devam)

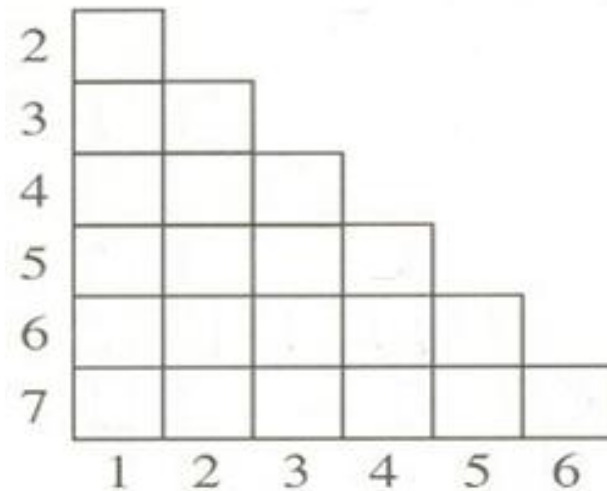
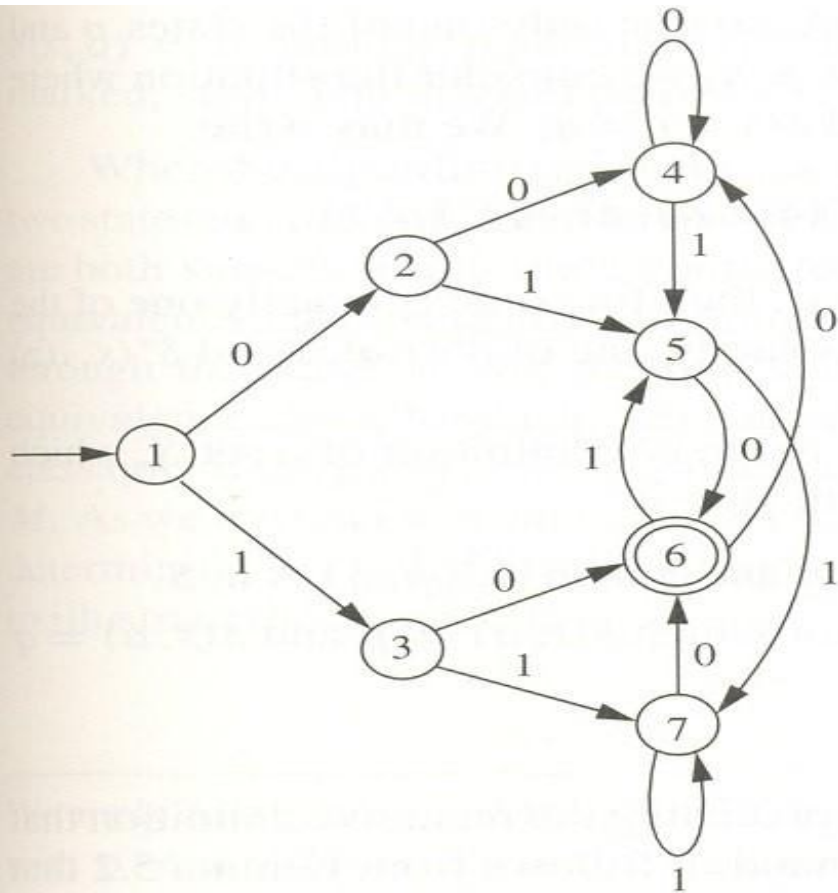
- States kümesinin partition kümesi, $(\{A, E\}, \{B, H\}, \{C\}, \{D, F\}, \{G\})$ olarak oluşturulur.
- Denk durumlar transitive'dir. **$p-q$ denk ise ve $q-r$ denk ise $p-r$ 'de denktir.**
- Partition kümesindeki her bir eleman bir durum olarak oluşturulur.
- Başlangıç durumu $\{A, E\}$ dir. Çünkü A orijinal otomatta başlangıç durumudur.
- C final state'dir. Çünkü orijinal otomatta final state'dir.
- Yeni oluşturulan her bir durumun tüm girişler için geçişleri düzenlenir.
- Örnek: $\{A, E\}$ 0 için $\{B, H\}$ 'ye geçer. Orijinal otomatta A 0 için B 'ye E ise H 'ye geçer. $\{A, E\}$ 1 için $\{D, F\}$ 'ye geçer. Orijinal otomatta A 1 için F 'ye ve E 1 için F 'ye geçer.



State Minimization

Örnek:

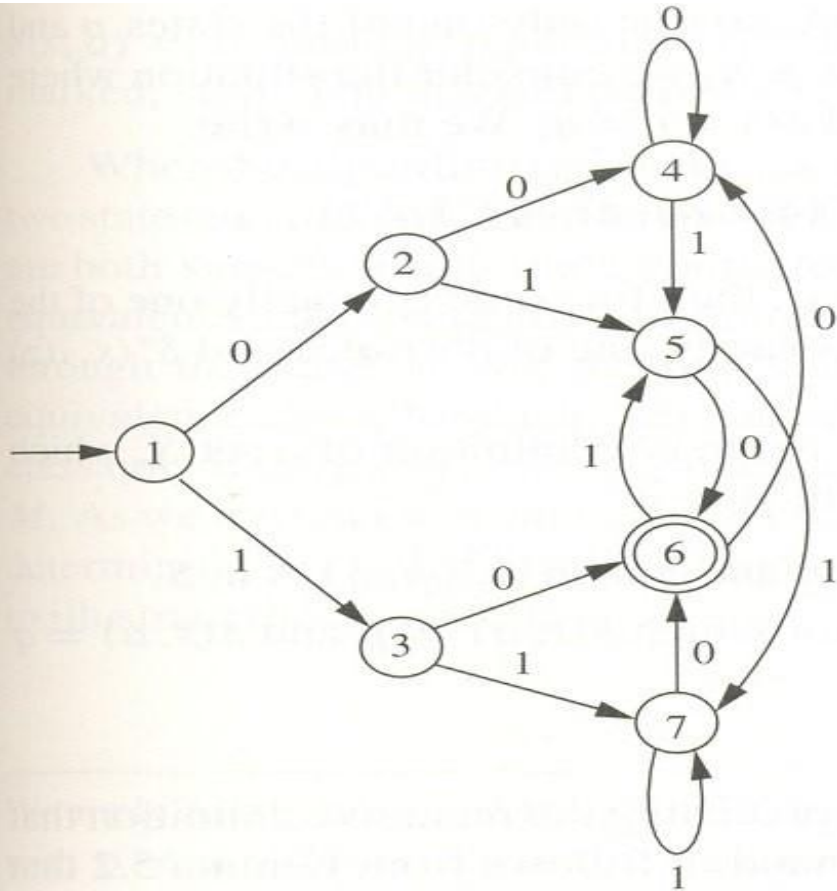
- Aşağıdaki otomata denk minimum duruma sahip otomatı bulunuz.



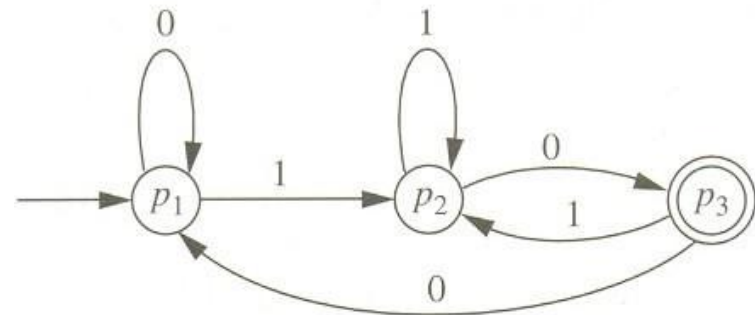
State Minimization

Örnek:

- Aşağıdaki otomata denk minimum duruma sahip otomatı bulunuz.



2						
3	2	2				
4			2			
5	2	2		2		
6	1	1	1	1	1	
7	2	2		2		1
	1	2	3	4	5	6



Ödev

- Problemleri çözünüz 2.5.3 (sayfa 101)

