(GNFA)

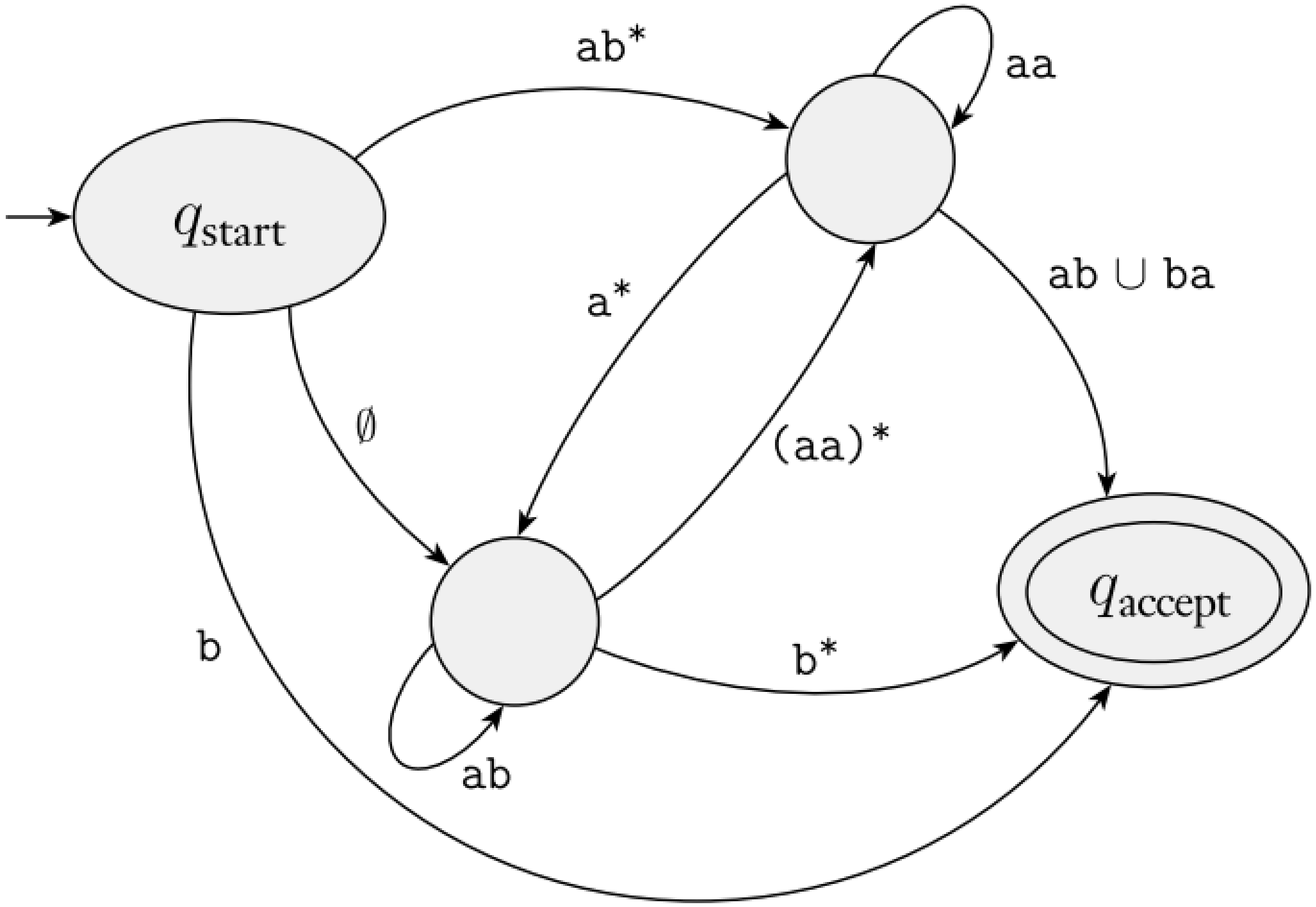
Genelleştirilmiş deterministik olmayan sonlu otomat (GNFA) adı verilen yeni bir sonlu otomat türünü tanımlayalım.

İlk önce DFA'ların GNFA'lara ve ardından GNFA'ların düzenli ifadelere nasıl dönüştürüleceğini gösteriyoruz.

Genelleştirilmiş deterministik olmayan sonlu otomatlar, basitçe deterministik olmayan sonlu otomatlardır; burada geçiş okları, yalnızca alfabenin veya ε'nin üyeleri yerine etiket olarak herhangi bir düzenli ifadeye sahip olabilir.

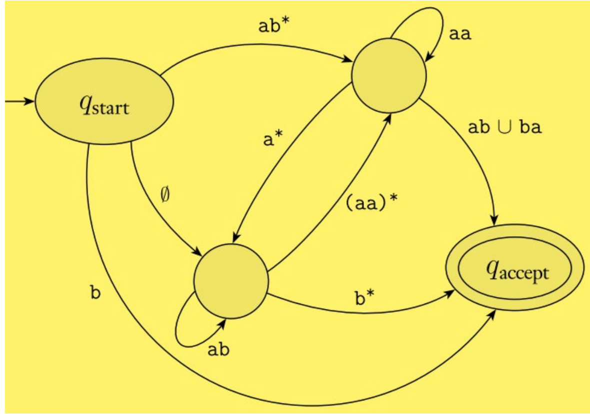
GNFA, sıradan bir NFA'da olduğu gibi her seferinde yalnızca bir sembol olmak zorunda değildir. Girdiden sembol bloklarını okur ve iki durumu birbirine bağlayan bir geçiş oku boyunca hareket eder. Bu semboller, o ok üzerindeki düzenli ifadeyle tanımlanan bir diziyi oluşturur

Bir GNFA deterministik değildir ve bu nedenle aynı giriş dizesini işlemek için birkaç farklı yola sahip olabilir. Eğer işlenmesi GNFA'nın girdinin sonunda kabul durumunda olmasına neden olabiliyorsa, katarı girdisini i kabul eder.



Generalized nondeterministic finite automaton

GNFA’lar aşağıdaki tanımlanacaktır:



• Başlangıç durumunda diğer tüm durumlara giden geçiş okları vardır ancak başka herhangi bir durumdan gelen ok yoktur

• Yalnızca tek bir kabul durumu vardır ve diğer tüm durumlardan oklar gelir, ancak buradan herhangi bir duruma giden başka ok yoktur. Ayrıca başlangıç durumu kabul durumu değildir.

• Başlangıç ve kabul durumları dışında, bir ok her durumdan diğer duruma ve her durumdan kendisine gider.

DFA'yı GNFA'ya dönüştürme

Basitçe eski başlangıç durumuna ε oklu yeni bir başlangıç durumu ve eski kabul durumlarından ε oklu yeni bir kabul durumu ekliyoruz.

Herhangi bir okun birden fazla etiketi varsa (veya aynı iki durum arasında aynı yönde giden birden fazla ok varsa), her birinin yerini, etiketi önceki etiketlerin birleşimi olan tek bir okla değiştiririz.

Son olarak, üzerinde ok bulunmayan durumlar arasına ∅ etiketli oklar ekliyoruz. Bu son adım, tanınan dili değiştirmeyecektir çünkü ∅ ile etiketlenmiş bir geçiş hiçbir zaman kullanılamaz.

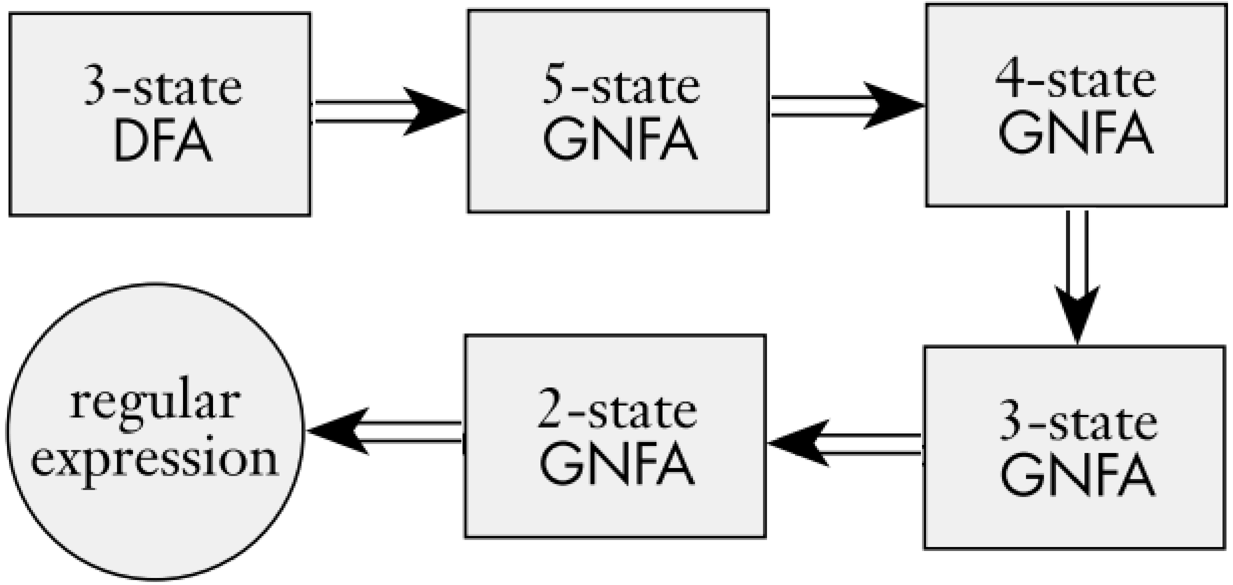
Buradan itibaren tüm GNFA'ların özel formda olduğunu varsayıyoruz.

Şimdi bir GNFA'nın regüler ifadeye nasıl dönüştürüleceğini gösteriyoruz. GNFA'nın k durumu olduğunu varsayalım. O halde, bir GNFA'nın bir başlangıç ve kabul durumunun olması ve bunların birbirlerinden farklı olması gerektiğinden, $k \geq 2$ olacaktır.

Eğer $k > 2$ ise, $k - 1$ durumlu eşdeğer bir GNFA oluştururuz. Bu adım, yeni GNFA'da iki duruma düşene kadar tekrarlanabilir.

Eğer $k = 2$ ise, GNFA'nın başlangıç durumundan kabul durumuna giden tek bir oku vardır. Bu okun etiketi eşdeğer regüler ifadedir.

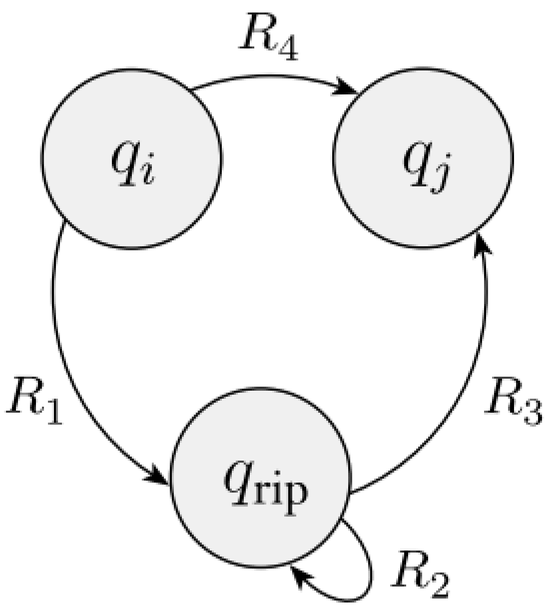
3 durumlu bir DFA'nın eşdeğer bir düzenli ifadeye dönüştürülmesinin aşamaları

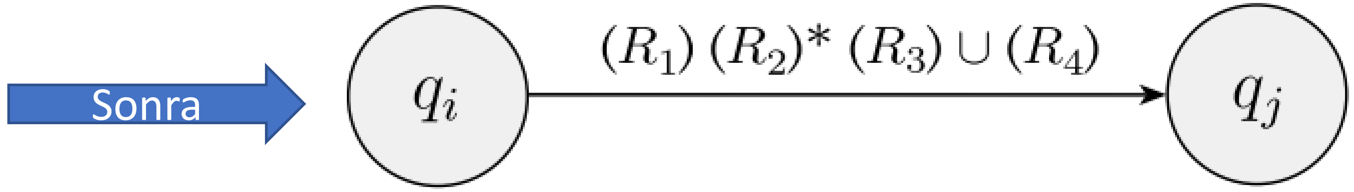


k > 2 olduğunda k-1 duruma sahip eşdeğer bir GNFA oluşturma

Bir durum seç ve onu makineden çıkar. Geri kalanı aynı dili tanıyacak biçimde düzenle.
Başlangıç veya kabul etme durumu olmaması koşuluyla herhangi bir durum bunu yapacaktır.
k > 2 olduğundan böyle bir durumun var olacağı garantidir. Kaldırılan duruma «qrip» diyelim.

qrip'i kaldırdıktan sonra, kalan okların her birini etiketleyen düzenli ifadeleri değiştirerek makineyi düzenliyoruz.
Yeni etiketler, kayıp hesaplamaları geri ekleyerek qrip eksikliğini telafi ediyor.
Qi durumundan qj durumuna giden yeni etiket, makineyi doğrudan veya qrip aracılığıyla qi'den qj'ye götürecek tüm dizeleri tanımlayan düzenli bir ifadedir.





Bu değişikliği, qi = qj durumu da dahil olmak üzere, herhangi bir qi durumundan herhangi bir qj durumuna giden her ok için yaparız. Yeni makine orijinal dili tanır.

Biçimsel olarak bir GNFA, aşağıdaki geçiş fonksiyonu ile ifade edilebilir.

$$\delta\colon (Q - \{q_{\text{accept}}\}) \times (Q - \{q_{\text{start}}\}) \longrightarrow \mathcal{R}.$$

Yani, Bir ok her durumu diğer tüm durumlara bağlar, ancak KABUL DURUMUNDAN gelen veya BAŞLANGIÇ DURUMUNA giden hiçbir ok yoktur.

A ***generalized nondeterministic finite automaton*** is a 5-tuple, $(Q, \Sigma, \delta, q_{\text{start}}, q_{\text{accept}})$, where

1. Q is the finite set of states,
2. Σ is the input alphabet,
3. $\delta\colon (Q - \{q_{\text{accept}}\}) \times (Q - \{q_{\text{start}}\}) \longrightarrow \mathcal{R}$ is the transition function,
4. q_{start} is the start state, and
5. q_{accept} is the accept state.

Bir GNFA, eğer $w = w_1w_2 \cdots w_k$ ise, Σ^* cinsinden bir w dizisini kabul eder,

burada her bir w_i katarı Σ^* içerisindedir ve $q_0, q_1,...,q_k$ durumlarının bir katarı mevcuttur;

1.$q_0 = q_{start}$ başlangıç durumudur,

2. $q_k = q_{accept}$, kabul durumudur ve

3. Her i için $w_i \in L(R_i)$ var. ($R_i = \delta(q_{i-1}, q_i)$; diğer bir deyişle R_i, q_{i-1}'den q_i'ye giden okun üzerindeki ifadedir.

M makinesi, A dili için bir DFA olsun. **Yeni bir başlangıç durumu** ve **yeni bir kabul durumu** ve gerektiği şekilde ek geçiş okları ekleyerek M'yi G isimli GNFA'ya dönüştürürürz. Bir GNFA'yı giriş olarak alan ve eşdeğer bir regüler ifade döndüren özyinelemeli CONVERT(G) algoritması tasarlarız. GNFA'nın iki durumunun olduğu durum özyinelemesiz çözülür.

CONVERT(G):

1. G'nin durum sayısı k olsun.

2. Eğer k = 2 ise, G bir başlangıç durumu, bir kabul durumu ve bunları birbirine bağlayan ve R düzenli ifadesiyle etiketlenmiş tek bir oktan oluşmalıdır. R ifadesini döndürün.

3. Eğer k > 2 ise, q_{start} ve q_{accept}'ten farklı herhangi bir qrip ∈ Q durumunu seçeriz. G''nin GNFA olsun (Q′ , Σ, δ′ , q_{start}, qaccept),

Q′ = Q − {q_{rip}},

Herhangi bir q_i ∈ Q′ − {q_{accept}} ve q_j ∈ Q′ − {q_{start}} için

$$\delta'(q_i, q_j) = (R_1)(R_2)^*(R_3) \cup (R_4)$$

$$R_1 = \delta(q_i, q_{\mathrm{rip}})$$

$$R_2 = \delta(q_{\mathrm{rip}}, q_{\mathrm{rip}})$$

$$R_3 = \delta(q_{\mathrm{rip}}, q_j)$$

$$R_4 = \delta(q_i, q_j)$$

4. CONVERT(G′)'yi hesaplayın ve bu değeri döndürün.

Temel adım: k = 2 durum için çözüm. Eğer G'nin yalnızca iki durumu varsa, başlangıç durumundan kabul durumuna giden yalnızca tek bir oka sahip olabilir. Bu oktaki düzenli ifade etiketi, G'nin kabul durumuna geçmesine izin veren tüm katarları ifade eder. Dolayısıyla bu ifade G'ye eşdeğerdir.

Tümevarım adımı: k - 1 durum için çözüm. G ve G’'nin aynı dili tanıyacaktır.

$$q_{\text{start}}, q_1, q_2, q_3, \ldots, q_{\text{accept}}.$$

Eğer bunlardan hiçbiri kaldırılan durum qrip değilse, açıkça G’ de w'yi kabul eder. G' oklarını etiketleyen yeni düzenli ifadelerin her biri, bir bütünün parçası olarak önceki regüler ifadeyi içermektedir.

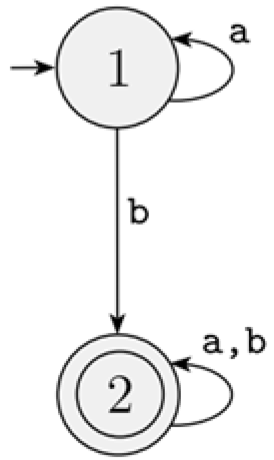
Eğer qrip ortaya çıkarsa, ardışık qrip durumlarının her bir çalışmasının kaldırılması, G' için kabul edilebilir bir hesaplama oluşturur. Bir koşuyu paranteze alan qi ve qj durumları, aralarındaki ok üzerinde, G'deki qrip aracılığıyla qi'yi qj'ye götüren tüm dizileri tanımlayan yeni bir düzenli ifadeye sahiptir. Dolayısıyla G’, w'yi kabul eder.

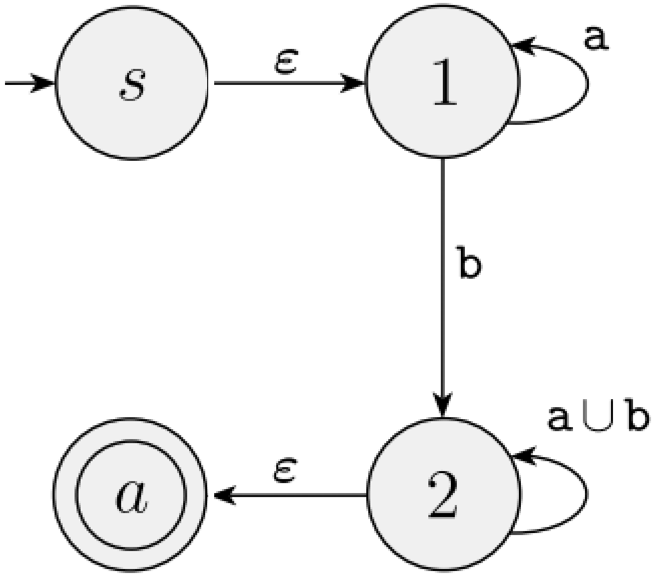
Eğer G, w katarını kabul ediyorsa, G'deki qi ve qj durumları arasındaki her ok, qi'yi doğrudan veya qrip yoluyla G'deki qj'ye götüren dizilerin toplamını tanımladığından, G'nin w'yi de kabul etmesi gerekir. Dolayısıyla G ve G' eşdeğerdir.

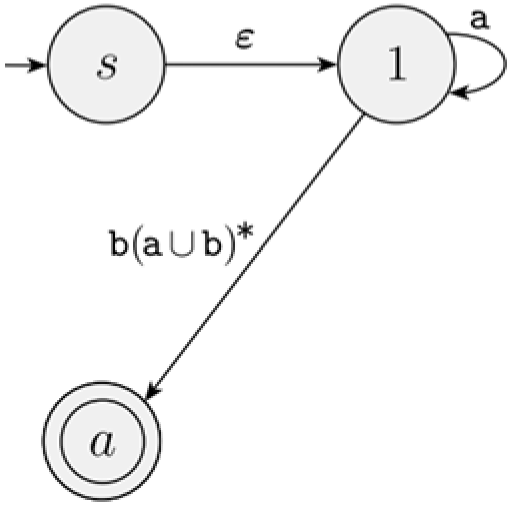
Algoritma kendisini G' girişinde yinelemeli olarak çağırdığında sonuç G'ye eşdeğer bir regüler ifadedir. G'nin k - 1 durumuna sahip olduğunu belirtir. Dolayısıyla bu regüler ifade de G'ye eşdeğerdir.

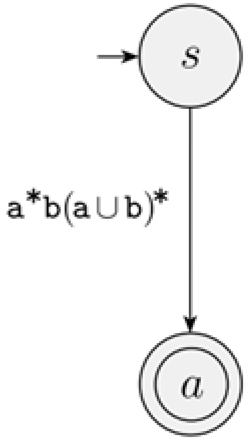
q$_{start}$ ve **q$_{accept}$** yerine s ve a adı verilen yeni bir başlangıç durumu ve yeni bir kabul durumu ekleyerek dört durumlu bir GNFA oluşturuyoruz, böylece bunları uygun şekilde çizebiliyoruz.

DFA'nın 2. durumundaki öz döngüdeki a, b etiketini, GNFA'nın karşılık gelen noktasındaki a∪b etiketiyle değiştiriyoruz. Bunu yapıyoruz çünkü DFA'nın etiketi biri a için, diğeri b için olmak üzere iki geçişi temsil ederken, GNFA'nın 2'den kendisine giden yalnızca tek bir geçişi olabilir.



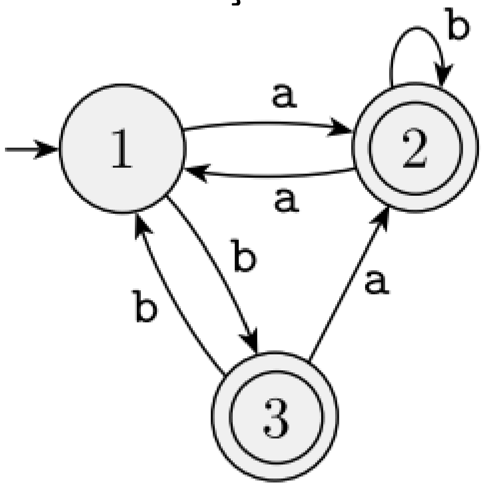




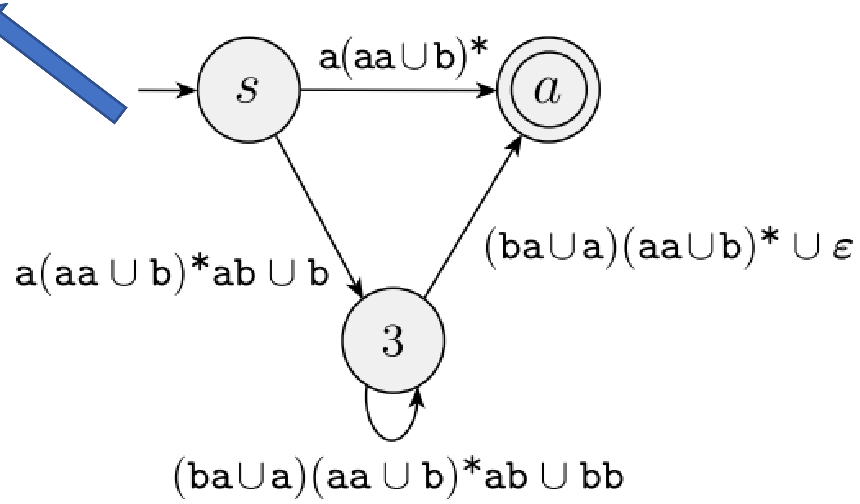


Durum 2'yi kaldırıyoruz ve kalan ok etiketlerini güncelliyoruz. Bu durumda değişen tek etiket 1'den a'ya olan etikettir.

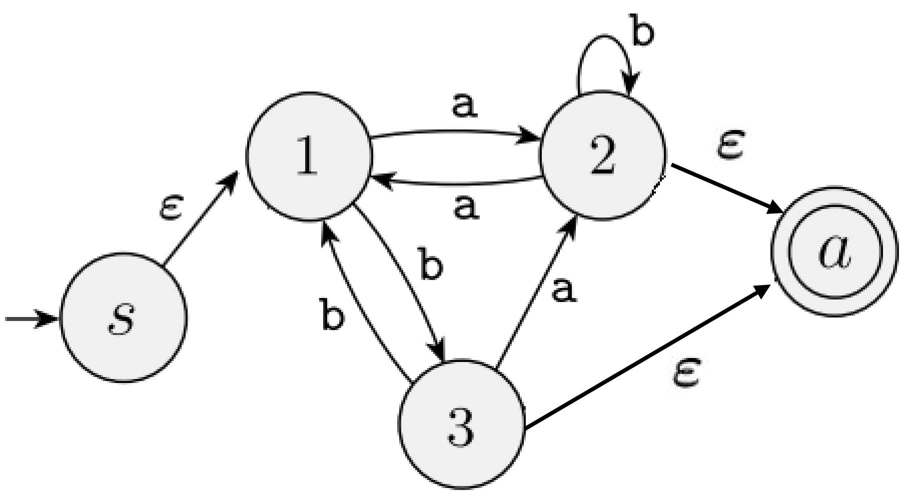
Örnek 2: Üçdurumlu bir DFA:

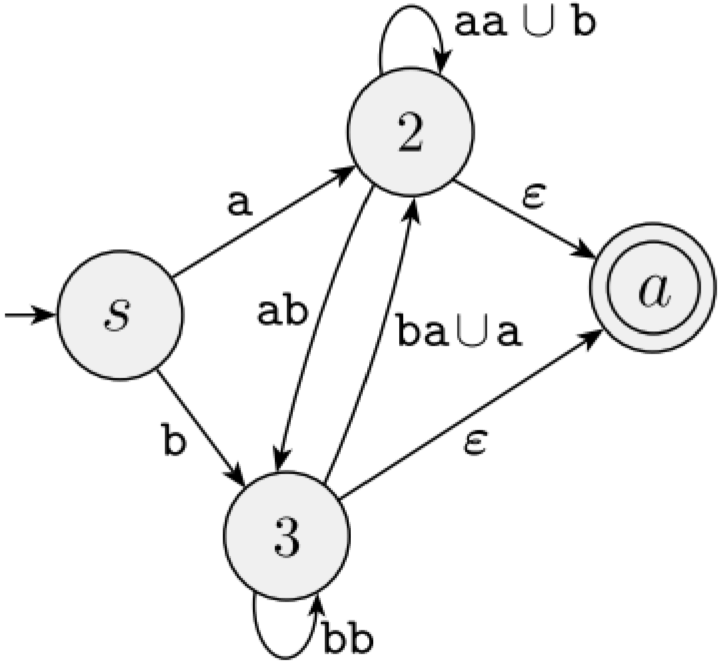


$$(\mathtt{a}(\mathtt{aa}\cup\mathtt{b})^*\mathtt{ab}\cup\mathtt{b})\,((\mathtt{ba}\cup\mathtt{a})(\mathtt{aa}\cup\mathtt{b})^*\mathtt{ab}\cup\mathtt{bb})^*((\mathtt{ba}\cup\mathtt{a})(\mathtt{aa}\cup\mathtt{b})^*\cup\varepsilon)\cup\mathtt{a}(\mathtt{aa}\cup\mathtt{b})^*$$



$$\delta\colon (Q-\{q_{\text{accept}}\})\times(Q-\{q_{\text{start}}\})\longrightarrow\mathcal{R}.$$





Bir sonraki ders

Regülerlik Testi

Pumping Lemma