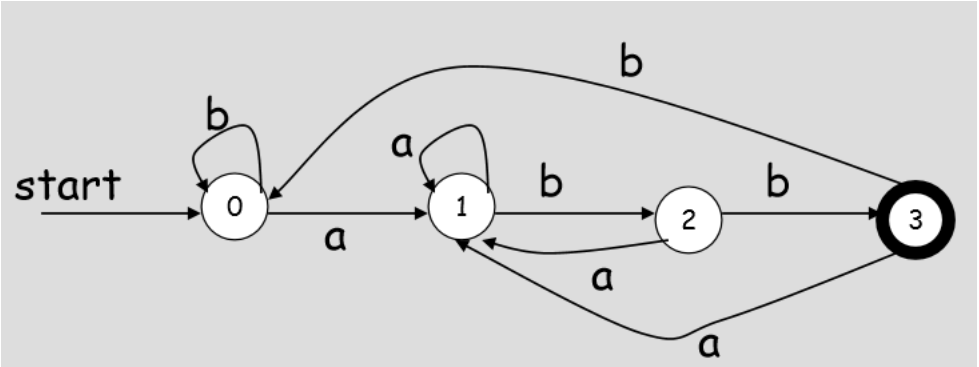
DFA – NFA

Otomata teorisi, matematiksel soyut makineler kullanılarak hesaplama problemlerinin çözülmesini araştıran bilgisayar bilimi dalıdır. Soyut makinelere **otomat** denir. Sonlu sayıda duruma sahip bir otomata **sonlu otomat** denir. Bir otomat durumlardan ve geçişlerden oluşur. Sembol dizesini girdi olarak alır ve durumunu buna göre değiştirir. İstenen sembol bulunduğunda geçiş gerçekleşir.

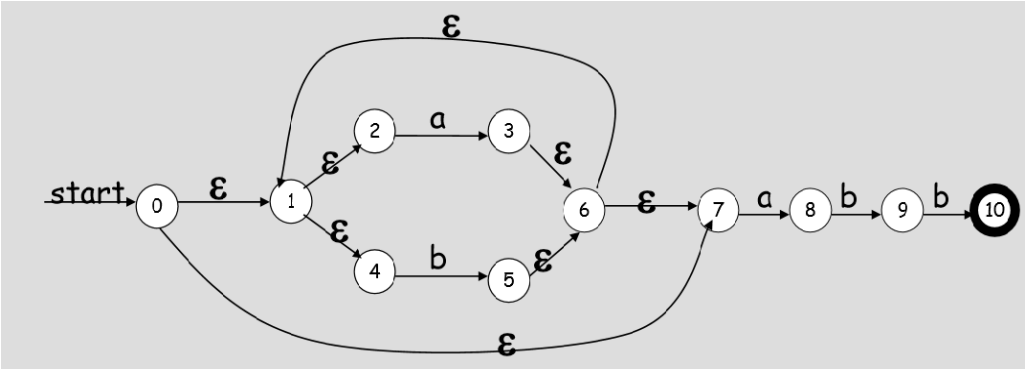
Deterministik Sonlu Otomata (DFA) ve Deterministik Olmayan Sonlu Otomata (NFA) olmak üzere iki tür sonlu otomata vardır.

1) **DFA (Deterministic Finite Automata):** DFA, sonlu otomatların özel bir halidir. Deterministik, hesaplamanın benzersizliğini ifade eder. DFA'da, geçerli durumdan bir sonraki duruma kadar belirli bir girdi için yalnızca bir yol vardır. DFA boş geçişi (ε) kabul etmez, yani DFA herhangi bir giriş karakteri olmadan durumu değiştiremez. DFA ile daha hızlı tanıma yapılırken daha fazla alan gerektirir. DFA'da geri izlemeye izin verilir.



Şekil 1. (a | b) * abb için DFA

2) **NFA (Non-Deterministic Finite Automata):** NFA, DFA'nın daha genelleştirilmiş halidir. Belirli bir girdi için mevcut durumdan bir sonraki duruma kadar birçok yol vardır. Boş geçişi (ε) kabul edebilir. NFA oluşturmak daha kolaydır ve aynı zamanda daha az alan gerektirir. Avantajı esnek olmasıdır. NFA'da geri izleme her zaman izin verilmez. Bazı durumlarda mümkün olmakla birlikte, bazılarında mümkün değildir. Her NFA makinesi için eşdeğer bir DFA vardır.



Şekil 2. (a | b) * abb için NFA

DFA – NFA KARMAŞIKLIK

Durum karmaşıklığı (State complexity); DFA ve NFA gibi sonlu otomata çeşitleri gibi soyut otomatların büyüklüğü ile ilgilenen teorik bilgisayar bilimi alanıdır. Sonlu bir otomatın durumu, otomatın belleğinde saklanan belirli bilgileri temsil eder. Bu nedenle, sonlu otomatanın durum karmaşıklığı Turing makinelerinin uzay karmaşıklığının bir benzetmesi olarak görülebilir.

DFA ve NFA her ikisi de aynı grup dilleri kabul ederler. Bu dillere düzenli diller denir. Thompson yapısı, düzenli ifadelerden NFA oluşturmak için kullanılan birkaç algoritmadan biridir. Algoritma, bir ifadeyi, NFA'nın bir dizi kural kullanılarak oluşturulacağı kurucu alt ifadelerine bölerek yinelemeli olarak çalışır. Bu kurallar; ε geçişleri, başlangıç durumu ve sembol kabul etme durumu, düzenli ifadelerin birleştirilmesidir. Thompson yapısında düzenli ifade karakteri başına en fazla iki yeni durum ve en fazla dört geçiş vardır. Bu kurallar ile elde edilen NFA yapım süresi O(m)'dir. O(m) zamanında tek bir karakter için durum geçiş kümesi hesaplanabilir. Bu nedenle NFA'nın çalışma süresi n uzunluğunda bir dize için O(nm)'dir. Yani NFA'nın çalışma süresi hem NFA boyutu hem de giriş dizesinin boyutu ile alakalıdır.

DFA durumları NFA durum kümelerinden oluştuğu için, bir n durumlu NFA en fazla 2^n durumlu bir DFA'ya dönüştürülebilir. O(2^n) en kötü durum(worst case) karmaşıklığını verir. En iyi durum, NFA'nın başlangıç durumunun başka herhangi bir duruma giden geçişi olmamasıdır. Yani aynı durumu DFA üzerinde göstermek daha maliyetlidir. Fakat DFA, NFA'ya göre daha hızlıdır. DFA'nın çalışma süresi n uzunluğunda bir dize için O(n)'dir. Çünkü belirli bir dize için DFA'da yalnızca bir yol vardır. Yani DFA'nın çalışma süresi yalnızca giriş dizesinin boyutuna bağlıdır.

	NFA	DFA
Build time	$O(m)$	$O(2^m)$
Run time	$O(m \times n)$	$O(n)$

m = length of regular expression
n = length of input string

Şekil 3. DFA,NFA karmaşıklığı

NFA'yı DFA'ya dönüştürmenin zaman karmaşıklığı ise O (n ^ 3 2 ^ n)'dir.

Her durum için epsilon kapanışlarının hesaplanması O (2^n) ve n durumları için hesaplanması O(n^3) gerektirir. Eş değer DFA'ya hesaplamak için ise O (2^n) durum gerekir.

Benzer şekilde, DFA'yı normal ifadeye dönüştürmenin zaman karmaşıklığı O (n ^ 3 4 ^ n)'dir.

Buse KARA 160401006

Nurşah DAĞLI 160401054