

CENG 306 Biçimsel Diller ve Otomatlar

Formal Languages and Automata

TURING MACHINE

(III)

Turing Machines

Tanım: Karmaşık Turing makineleri basit makinelerin birleşimi şeklinde oluşturulabilir.

Basit Makineler:

Symbol-writing machines (sembol yazma makineleri):

Her $a \in \Sigma \cup \{\leftarrow, \rightarrow\} - \{\Delta\}$ için $M_a = (\{s, h\}, \Sigma, \delta, s, \{h\})$ tanımlanabilir ve $\delta(s, b) = (h, a)$ olur, burada $b \in \Sigma \cup \{\Delta\}$.

Burada da $\delta(s, \Delta) = (s, \rightarrow)$ otomatik geçişi geçerlidir.

Bu makine sadece a işlemini yapar. Eğer $a \in \Sigma$ ise a yazılır, $a \in \{\leftarrow, \rightarrow\}$ ise sola veya sağa gidilir ve makine durur.

Yazma makinesi çok sık kullanılacağı için M_a yerine kısaca \mathbf{a} kullanılır.

Head-moving machines:

M_{\leftarrow} ve M_{\rightarrow} şeklindedir ve kısaca \mathbf{L} (left) ve \mathbf{R} (right) makinesi olarak gösterilir.

Turing Machines

Tanım: Karmaşık TM'ler basit TM'leri (a, L, R) birleştirilerek oluşturulabilir.

Makine birleştirme kuralları:

- Makineler FA'daki durumlar gibidir ve durumların bağlanması şeklinde birleştirilir
- Bir makineden diğerine yapılan **bağlantı** ilkinin **halt durumuna geçmesiyle** çalışır ve ikinciye geçilir.
- İkinci makine başlangıç durumuyla çalışmaya başlar.

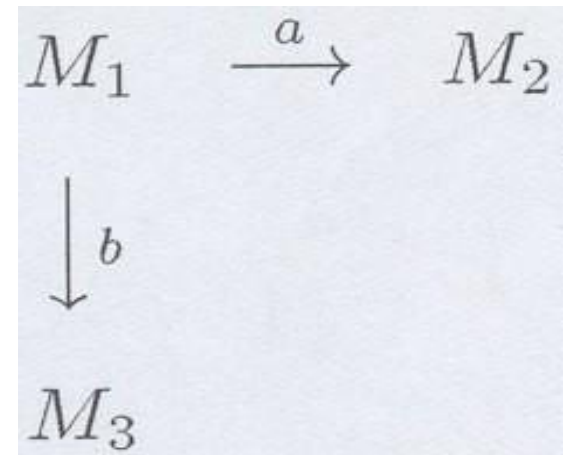
Örnek: Yandaki şekilde M_1 , M_2 ve M_3 Turing makinesidir.

M_1 başlangıç durumunda çalışmaya başlar.

M_1 halt durumuna geçince okunan sembol:

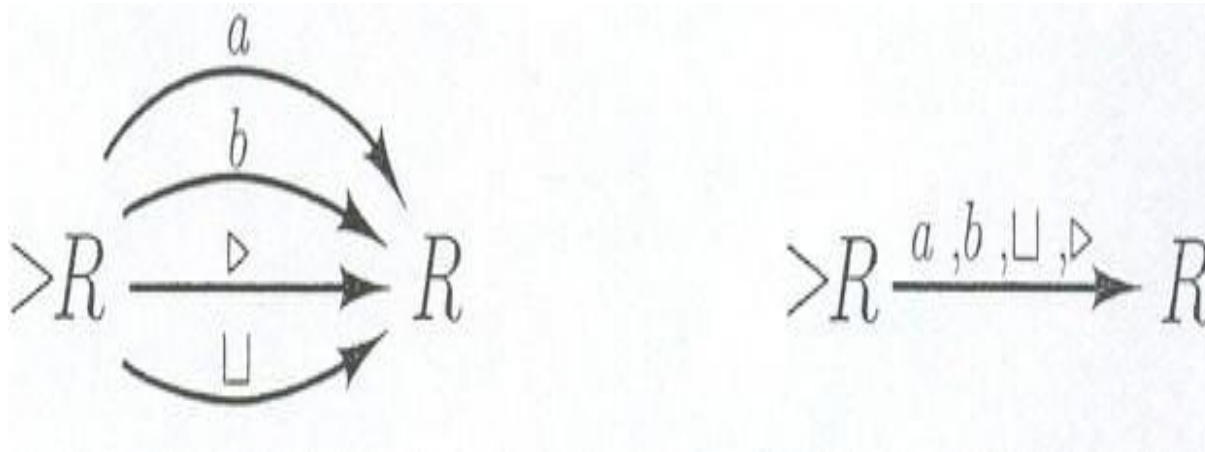
a ise M_2 başlangıç durumunda çalışmaya başlar,

b ise M_3 başlangıç durumunda çalışmaya başlar.



Turing Machines

Örnek: iki R makinesi aşağıdaki gibi birleştirilsin.



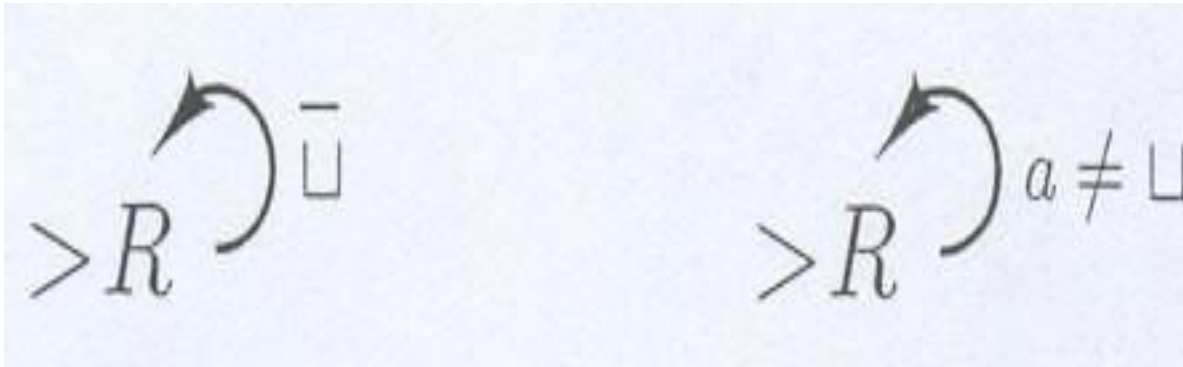
- Bu makine okuma kafasını önce bir sağa geçirir ve okunan sembol a , b , Δ veya \sqcup ise bir sağa daha geçirir.
- Eger bir geçiş alfabedeki tüm sembolleri içerirse etiket yazılmadan $R \rightarrow R$ şeklinde gösterilir. Daha da basitleştirilerek ise

RR veya R^2

şeklinde gösterilebilir.

Turing Machines

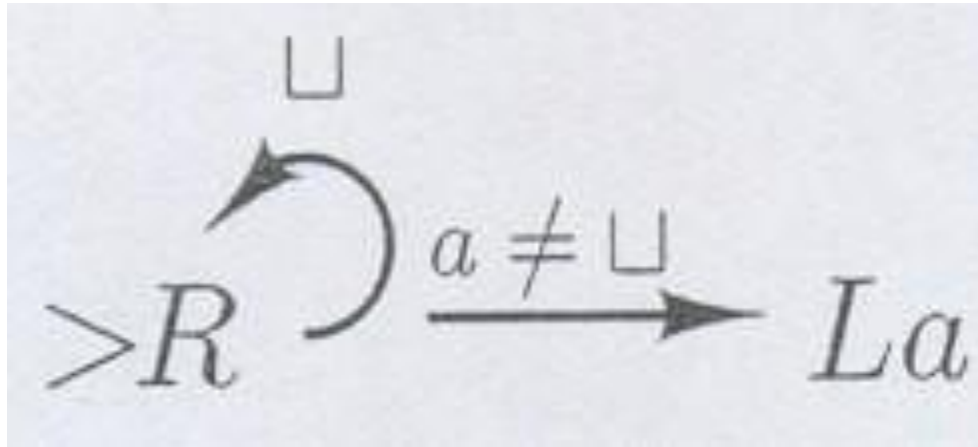
Örnek: Eger $a \in \Sigma$ ise, birçok sembol kullanılan oklar yerine a şeklinde gösterim de kullanılabilir.



- Yukarıdaki şekilde soldaki makine \sqcup bulana kadar sağa gider ve R_{\sqcup} şeklinde gösterilir.
- Yukarıdaki sağdaki şekil de aynı işlemi ifade etmektedir. Ancak okunan a (sembolik ifade) sembolünün daha sonra kullanılmasını sağlamaktadır.

Turing Machines

Örnek: Aşağıdaki makine bir sembol bulana kadar sağa gider ve bulduğu sembolü bir soldaki alana kopyalar.

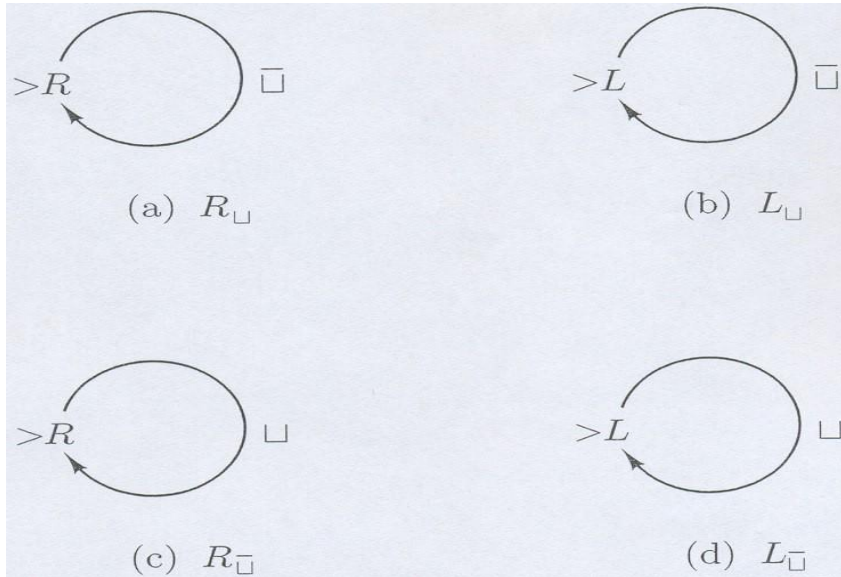


La bir sola gitmeyi ve en son okunan $a \in \Sigma$ sembolünü yazmayı ifade etmektedir.

Turing Machines

Örnek: Aşağıdaki makineler hep sağa veya sola gider ve bir sembol arar.

Aradığını bulur bulmaz çalışması sonlanır.



(a) R_{\square} , sağa doğru tarama yapar ve ilk bulduğu boşlukta durur.

(b) L_{\square} , sola doğru tarama yapar ve ilk bulduğu boşlukta durur.

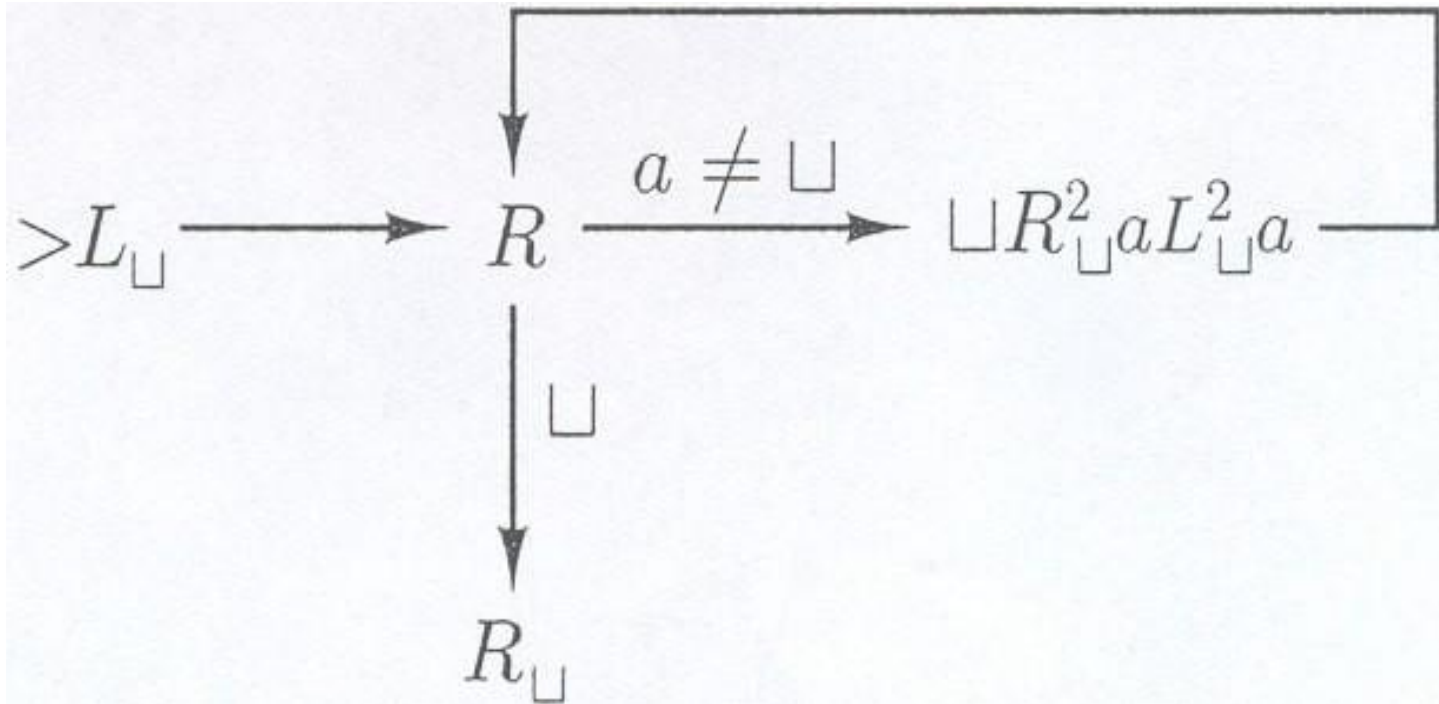
(c) R_{\sqcup} , sağa doğru tarama yapar ve ilk bulduğu sembolde durur.

(d) L_{\sqcup} , sola doğru tarama yapar ve ilk bulduğu sembolde durur.

Turing Machines

Örnek: Aşağıdaki kopyalama makinesi bir w stringini sağ kısmına kopyalar.

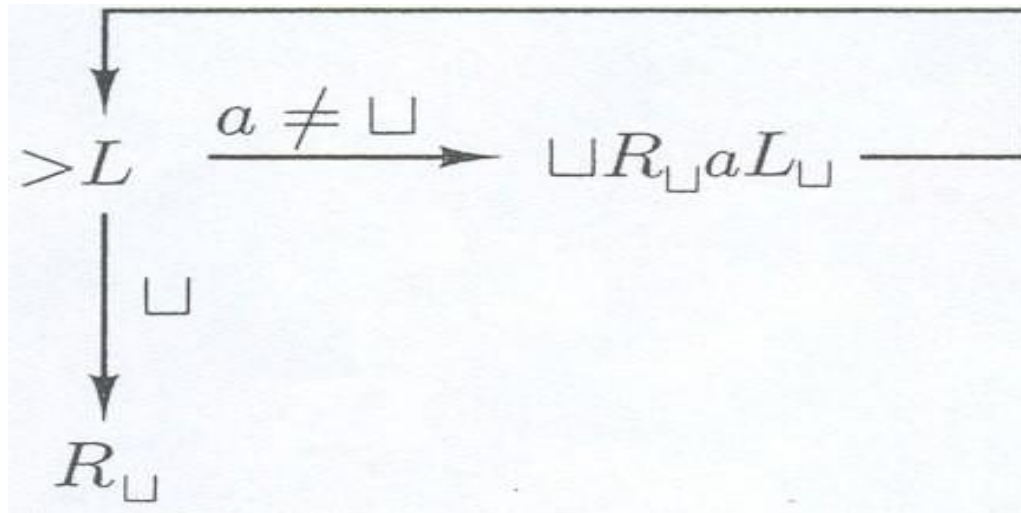
$\square w \square$ string'i için sonuç string'i $\square w \square w \square$ olur.



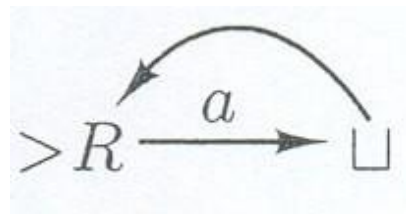
Turing Machines

Örnek: Aşağıdaki sağa kaydırma makinesi bir w stringini bir sağa kaydırır.

$\square w \square$ string'i için sonuç string'i $\square \square w \square$ olur.



Örnek: Aşağıdaki makine tape üzerindeki tüm a ları siler.



Computing with Turing Machines

- *Turing makinesinde işlemler için gerekli giriş string'i Δ sembolünün sağına yazılır ve içinde boşluk sembolü yoktur.*
- *Giriş string'inin sağındaki kısım tümüyle boşluk sembolüdür.*
- *Bundan sonraki örneklerde giriş string'i ile Δ sembolü arasında bir boşluk sembolü (\sqcup) vardır.*
- *Eğer $M = (K, \Sigma, \delta, s, H)$ bir Turing makinesi ve $w \in (\Sigma - \{\sqcup, \Delta\})^*$ ise M makinesinin w girişi için başlangıç konfigürasyonu $(s, \Delta \sqcup w)$ şeklindedir.*

Computing with Turing Machines

Tanım : $M = (K, \Sigma, \delta, s, H)$ bir Turing makinesi

$$H = \{y, n\}$$

şeklinde iki tane halt state'e sahip olsun. Bunlar

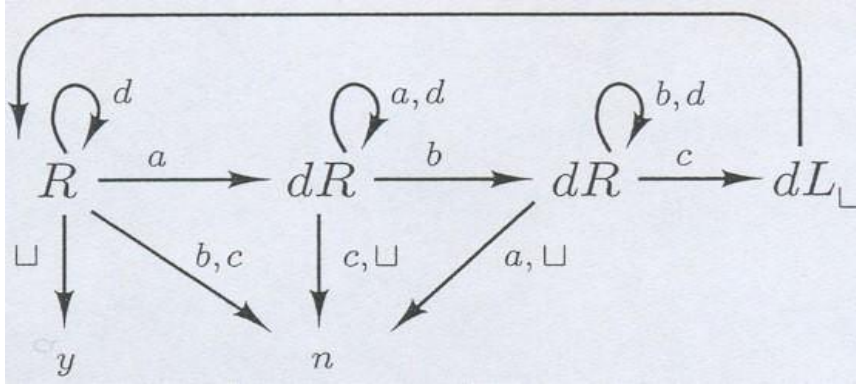
y kabul konfigürasyonu (**accepting configuration**) ve

h red konfigürasyonu (**rejecting configuration**) olsun.

- Eger $w \in (\Sigma - \{\sqcup, \Delta\})^*$ girişi için $(s, \Delta \sqcup w)$ konfigürasyonu M makinesini accepting configuration'lardan birisinde sonlandırırsa w bu dile aittir, halting configuration'lardan birisinde sonlandırırsa dile ait değildir.
- M makinesi bir dili belirler (**decide**) ve
 - eğer $w \in L$ ise w string'ini kabul eder
 - eğer $w \notin L$ ise w string'ini red eder
- Bir dili belirleyen bir TM varsa bu dil özyineliemeli (**recursive**) olarak adlandırılır.

Computing with Turing Machines

Örnek : $L = \{a^n b^n c^n : n \geq 0\}$ dilini tanıyan Turing makinesi aşağıdadır.



- *M makinesi n döngü yapar. Her döngüde makine, girişin en solundan başlar ve ilk bulduğu a yerine d , ikinci bulduğu b yerine d ve üçüncü olarak bulduğu c yerine d yazar.*
- *Okuma kafası yeniden string'in en soluna gider.*
- *Makine bir a ararken b veya c ye, b ararken c veya \sqcup 'ya, c ararken a veya \sqcup 'ya rastlarsa n durumuna gider.*
- *Eğer bir a ararken \sqcup gelirse (sağ kısmın tamamı d olmuştur) çalışmasını y durumuna geçerek sonlandırır.*

Computing with Turing Machines

Tanım : $M = (K, \Sigma, \delta, s, \{h\})$ bir Turing makinesi ve $\Sigma_0 \subseteq \Sigma - \{ \sqcup, \Delta \}$ ve $w \in \Sigma_0$ olsun.

Eğer M makinesi, w girişi için halt state'lere ulaş ıyorsa ve bazı y 'ler için

$$(s, \Delta \sqcup w) \vdash^*_M (h, \Delta \sqcup y) \text{ ise,}$$

y durumu M makinesinin çıkışı olarak adlandırılır ve $M(w)$ şeklinde gösterilir.

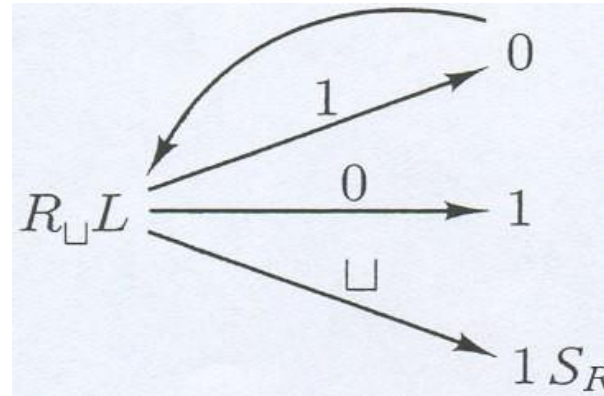
$M(w)$ sadece makinenin halt durumuna ulaşması halinde tanımlıdır.

Eğer bir f fonksiyonu Σ_0 dan Σ_0 a tanımlı ve tüm $w \in \Sigma_0$ için $M(w) = f(w)$ ise M makinesi f fonksiyonunu hesaplar.

M makinesinin çalışması bittiğinde tape üzerinde $\Delta \sqcup f(w)$ vardır ve bu fonksiyona özyinelemeli (**recursive**) denilir.

Computing with Turing Machines

Örnek : Aşağıda binary olarak yazılmış sayının bir fazlasını hesaplayan bir TM görülmektedir. ($\text{succ}(n) = n + 1$)



- *M makinesi önce girişin en sağıını bulur.*
- *Sonra 1 gördüğü sürece sola gider ve her 1 değerini 0 olarak değiştirir.*
- *ilk gördüğü 0 yerine 1 yazarak çalışmasını sonlandırır.*
- *Eğer sola giderken \square sembolü görürse yerine 1 yazar ve tüm girişi sağa bir pozisyon shift ederek çalışmasını sonlandırır.*

Computing with Turing Machines

Tanım: $M=(K, \Sigma, \delta, s, H)$ bir Turing makinesi, $\Sigma_0 \subseteq \Sigma - \{\sqcup, \Delta\}$

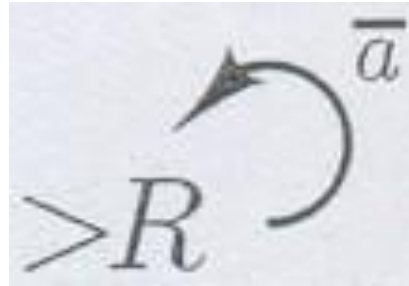
alfabe ve $L \subseteq \Sigma_0^*$ olsun.

0

- Eger sadece $w \in L$ iken M makinesi halt durumuna geçerse, M makinesi L dilini yarı belirler (**semidecides**) denir.
- Bir dil bir TM tarafından semidecide ediliyorsa bu dil özyinelemeli-sıralı (**recursively enumerable**) olarak adlandırılır.

Computing with Turing Machines

Örnek: $L = \{w \in \{a, b\}^* : w \text{ içinde en az bir } a \text{ vardır}\}$ şeklinde tanımlı bir dil aşağıdaki TM tarafından semidecide edilir.



- $w \in \{a, b\}^*$ girişi için makine $(q_0, \Delta \sqcup w)$ başlangıç konfigürasyonundan çalışmaya başladığında sağa doğru ilk a okuduğunda çalışmasını sonlandırır.
- Eger a bulamazsa sonsuza kadar çalışır ve hiçbir zaman halt durumuna ulaşamaz.
- M makinesi L dilini semidecide yapar ve L recursively enumerable dildir.