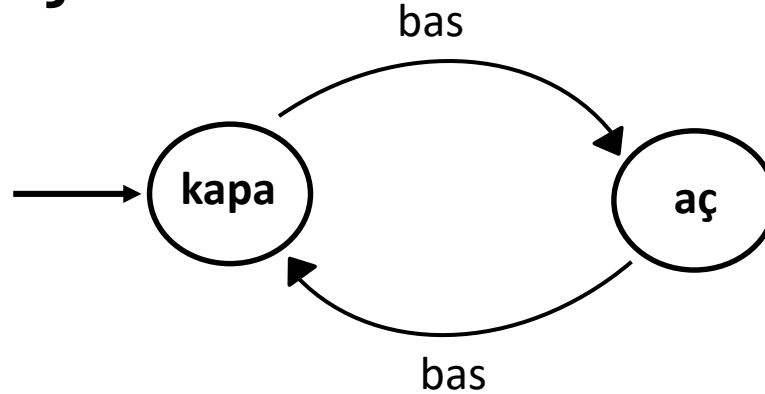


# Otomata Teorisi

## Fırat İsmailođlu, PhD

### Hafta I: Amaç ve Genel Kavramlar



# I. İletişim ve ders bilgisi

## Dersi sorumlusu:

Dr. Öğr. Üys. Fırat İsmailoğlu

email: [fismailoglu@cumhuriyet.edu.tr](mailto:fismailoglu@cumhuriyet.edu.tr)

Tel: 0346 210 1010 - 2462

Bilgisayar Müh. Bölümü, Oda No:212

## Ders Kitapları :

(1) Michael Sipser - *Introduction to the Theory of Computation (2nd Edition)* Thompson, 2005

(2) Anil Maheshwari, Michiel Smid - *Introduction to Theory of Computation*

## Puanlama:

%40 vize + %60 final (ödev verilmeyecek) (yaz okulu olmayacak)



## II. Otomata teorisi genel bakış:

Otomata ismi, Yunanca “*kendi başına hareket*” kelimesinden gelmektedir.

Otomata teorisi, hesaplamanın ve hesaplanabilirliğin prensiplerini anlamaya çalışır. Hatta bir diğer adı ‘Hesaplama Teorisi (Theory of Computation)’ dir.

Ne tür şeyleri mekanik olarak hesaplayabiliriz, hesaplayabildiğimiz şeyleri ne kadar hızlı hesaplarız, bunun için ne kadar hafızaya ihtiyacımız vardır sorularını sorar.

Otomata bu sorular için basit, soyut modeller ortaya koyar.

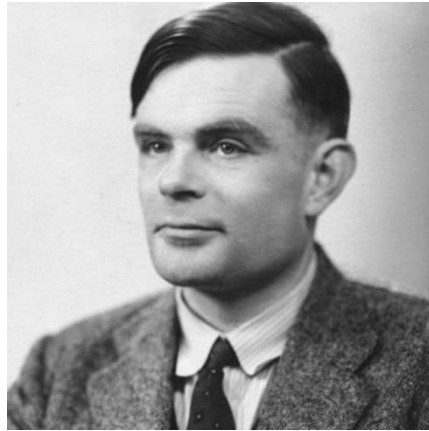
Bu modellere “*soyut makineler*” denir.



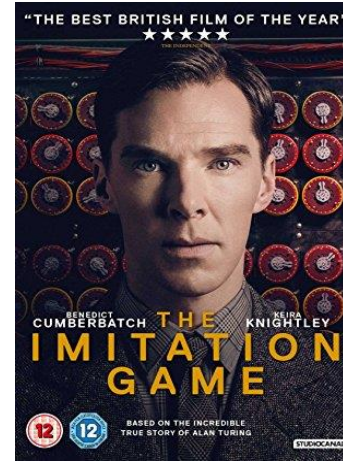
## II. Otomata teorisi genel bakış:



Bu alanda ilk çalışmalar **Alan Turing** tarafından 1930'lu yıllarda yapılmıştır ve Alan Turing'in geliştirdiği "Turing Makinesi" günümüz bilgisayarlarının atası kabul edilir.



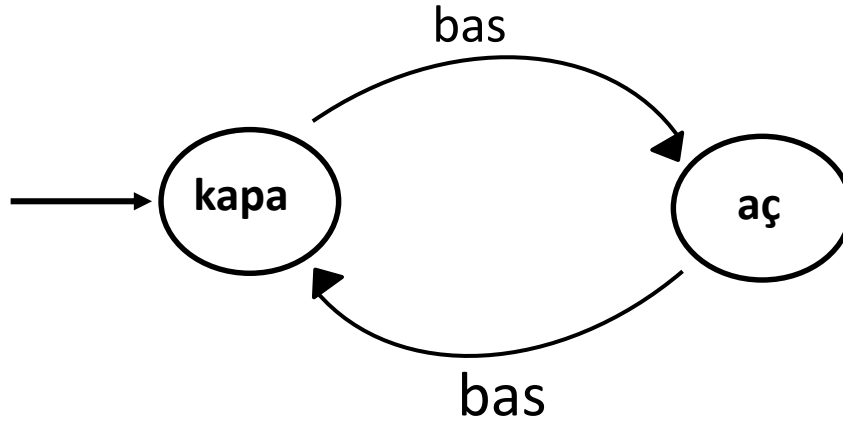
Alan Turing



### III. Hedeflenen Kazanımlar :

+ Problemleri soyutlayabilme, basitçe ifade edebilme

ör.



Ac/kapa düğmesi

+ Sınırlı kapasite ile çalışabilme deneyimi elde etme.

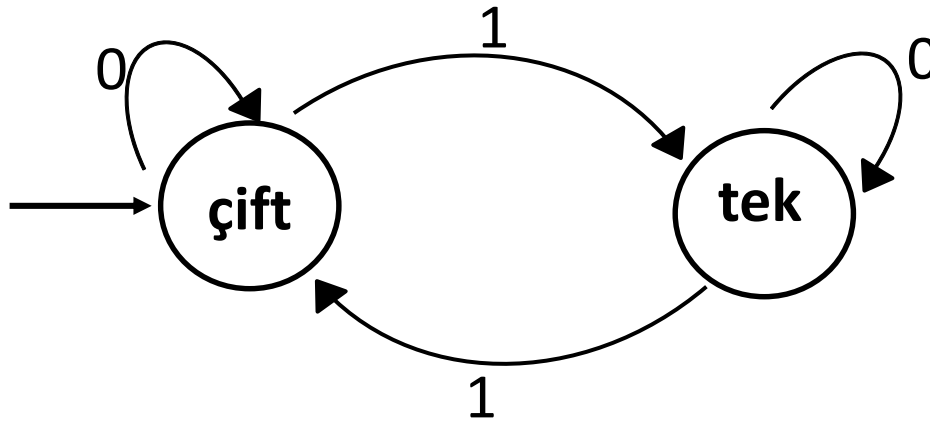
+ İlk bilgisayarların nasıl çalıştığı hakkında bilgi sahibi olma.

+ Bilgisayar mimarisi dersine temel oluşturma

### III. Hedeflenen Kazanımlar :

Sınırlı kapasite ile çalışabilme örneği:

Verilen 001001000....011 gibi büyük bir 0-1 dizisindeki toplam 1 sayısı tek mi diye kontrol etmek istiyoruz. Fakat makinemiz bu sayının tamamını belleğinde tutamıyor. Bu durumda aşağıdaki gibi basit bir makine bu görevi yerine getirebilir.






# Otomata için Genel Kavramlar

## Diller ve Kelimeler

**Alfabe** ( $\Sigma$ ): Sonlu, boş olamayan kümeye alfabe denir.

**Harf**: Alfabenin elemanlarına harf denir.

ör.  $\Sigma_1 = \{0,1\}$ ,  $\Sigma_2 = \{a, b, c, d\}$ ,  $\Sigma_3 = \{$      $\}$

**Kelime** (string) : Sonlu harf dizisi ör.  $k_1 = 01001$ ,  $k_2 = dddbcb$ ,  
 $k_3 =$    

Kelime uzunluğu  $| \cdot |$  ile verilir. ör.  $|k_1| = 5$

**Boş kelime** (empty string) : 0 uzunluğundaki kelime.  $\varepsilon$  ile gösterilir.  
Yani hiç harfi yoktur.

Kelimelerin bitleştirilmesi (concatenation): kelimelerin uç uca eklenmesi ile oluşur.

ör.  $k_1 = ba$  ve  $k_2 = na$  iken  $k_1 k_2 = bana$ ,  $k_1 k_1 = k_1^2 = baba$ ,  
 $k_1 k_2 k_2 = banana$ .

# Otomata için Genel Kavramlar

## Alfabelerin güçleri

$\Sigma$  bir alfabe iken  $\Sigma^k$ ,  $\Sigma$  alfabesinin harflerinden oluşan  $k$  uzunluğundaki kelimelerin kümesidir.

ör.  $\Sigma = \{0,1\}$  iken  $\Sigma^1 = \{0,1\}$ ,  $\Sigma^2 = \{00,01,10,11\}$ , ...

**Not:** Tüm  $\Sigma$  alfabeleri için  $\Sigma^0 = \{\varepsilon\}$ . Yani yalnızca boş kelimeyi içerir.

$\Sigma^*$ :  $\Sigma$  alfabesinin harfleri kullanılarak oluşturulabilecek bütün kelimelerin kümesidir:

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots$$

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





# Otomata için Genel Kavramlar

**Dil:** Bir  $\Sigma$  alfabeti üzerine bir dil  $\Sigma^*$  'ın bir alt kümesidir.  $L$  ile gösterilir.

**ör.**  $L = \{0,00,001,0000,00001, \dots\} \rightarrow \Sigma = \{0,1\}$  alfabeti üzerine 0 ile başlayan kelimelerin dili.

**ör.**  $L = \{\varepsilon, 01,10,0011,0101,1001, \dots\} \rightarrow \Sigma = \{0,1\}$  alfabeti üzerine içerdiği toplam 0 sayısı içerdiği toplam 1 sayısına eşit olan kelimelerin dili.

**ör.** Türkçe dili  $\Sigma = \{a, b, c, \dots, y, z\}$  üzerine kelimelerin kümesi.

**Dillerin bitleştirilmesi** (concatenation of languages) : Kelimelerin bitleştirilmesi kavramı diller için de genişletilebilir.  $L_1$  ve  $L_2$  iki dil iken bunların bitleştirilmesi

$$L_1L_2 = \{xy \mid x \in L_1, y \in L_2\}$$

**ör.**  $L_1 = \{0, 01\}, L_2 = \{\varepsilon, b, bb\}$  iken  $L_1L_2 = \{0,0b, 0bb, 01,01b, 01bb\}$



$$0\varepsilon = 0$$