CENG 114 BİLGİSAYAR BİLİMLERİ İÇİN AYRIK YAPILAR

Doç. Dr. Tufan TURACI
tturaci@pau.edu.tr

- Pamukkale Üniversitesi
- Mühendislik Fakültesi
- Bilgisayar Mühendisliği Bölümü

- Hafta 6

Ders İçereği

- **Boolean Fonksiyonlar**

- **Algoritma Analizi**

  --- Yürütme Zamanı

  --- Karmaşıklık

Boolean Fonksiyonlar

--- Boolean matematiği sayısal devrelerin çıkış ifadelerinin giriş değişkenleri cinsinden ifade edilmesi ve elde edilen ifadenin en basit haline ulaşması için kullanılmaktadır.

--- Sayısal olarak bir değişken veya fonksiyon iki değer alabilir. Bu değerler 1 veya 0 dır.

ÖRN: Halamda, kardeşimde, kedim ve köpeğimde bit var olup, olmaması olayı haftanın yedi günü için aşağıdaki şekilde verilmiştir.

	Pazartesi	Salı	Çarşamba	Perşembe	Cuma	C.tesi	Pazar
A: Halamda bit var	0	0	1	0	1	0	0
B: Kardeşimde bit var	0	0	1	0	0	0	1
C: Kedimde " "	0	1	1	0	0	0	1
D: Köpeğimde " "	1	1	1	0	1	0	1

Aşağıdaki önermelerin lojik ifadesini yazınız

i) Halamda ve kardeşimde bit var. $A.B = 0010000$

ii) Halamda bit var fakat kedimde yok $A.\overline{C} = 0000100$

iii) Halamın bitli fakat kardeşimin bitsiz veya kardeşimin bitli kedimin bitsiz veya kedimin bitli halamın bitsiz $\left.\right\}$ $A.\overline{B} + B.\overline{C} + C.\overline{A}$

$$= 0000100 + 0000000 + 0100001$$

$$= 0100101$$

salı, cuma, pazar

7\. ÖRN

Eğer alarm düğmesine basılı ve kapı kapalı değilse veya saat 6'dan sonra ve pencere kapalı değilse alarm çalar ifadesinin lojik fonksiyonu

A : Düğmeye basılı
B : Kapı kapalı
C : Saat 6'dan sonra
D : Pencere kapalı
F : Alarm çalar

$$F = A.\overline{B} + C.\overline{D}$$

Önemli Teoremler

$$Teo.1)\ x_1 + 0 = x_1$$

$$Teo.2)\ x_1 + 1 = 1$$

$$Teo.3)\ x_1 + x_1 = x_1$$

$$Teo.4)\ x_1 + \overline{x_1} = 1$$

$$Teo.5)\ x_1 . 0 = 0$$

$$Teo.6)\ x_1 . 1 = x_1$$

$$Teo.7)\ x_1 . x_1 = x_1$$

$$Teo.8)\ x_1 . \overline{x_1} = 0$$

$$Teo.9)\ x_1 + x_1 . x_2 = x_1$$

$$Teo.10)\ x_1 .(x_2 + x_3) = x_1 . x_2 + x_1 . x_3$$

$$Teo.11)\ x_1 + \overline{x_1} . x_2 = x_1 + x_2$$

$$Teo.12)\ x_1 .(x_1 + x_2) = x_1$$

$$Teo.13a)\ x_1 + x_2 = x_2 + x_1$$

$$Teo.13b)\ x_1 . x_2 = x_2 . x_1$$

$$Teo.14)\ \overline{x_1 + x_2} = \overline{x_1} . \overline{x_2}$$

$$Teo.15)\ \overline{x_1 . x_2} = \overline{x_1} + \overline{x_2}$$

$$Teo.16)\ \overline{\overline{x_1}} = x_1$$

Lojik **fonksiyon** ifadeleri değişkenlerin değil, ve, ya da işlemleri ile birleşmesinden ortaya çıkar.

Bu ifadeler 3 farklı grupta toplanır

① Doğal biçim: Bir değişken üzerine hiçbir işlem uygulanmamışsa bu değişkeni içeren ifadeye doğal biçim denir.

$$x_1 . x_2 + x_3, \quad \overline{x_1 . x_2} + x_3$$

② Yadonsal biçim

③ Vedensel biçim.

Yadonsal Tek Biçim — Vedensel Tek biçim
(minimal form) (maximal form)

Yadalama biçimindeki bir ifade üzerine "yada" işlemi uygulanmış her alt ifadeye esas ifadenin yadonsallığı denir.

Vedeleme biçimindeki bir ifadenin üzerine "ve" işlemi uygulanmış her alt ifadesine esas ifadenin vedenselliği denir.

$f(x_1, x_2, x_3) \to x_1 . x_2 . \overline{x_3} \Rightarrow$ (Vedensel)

$f(x_1, x_2, x_3) \to \overline{x_1} + x_2 + \overline{x_3} \Rightarrow$ (Yadonsal)

Yadonsal biçimindeki bir ifadenin herbir yadonsalı ortamdaki tüm değişkenlerin yada değillerinin "ve"lenmesi ile elde edilmiş ise bu ifadeye yadonsal tek biçim denir.

$$f(x_1,x_2,x_3) = x_1.x_2.x_3 + x_1.\bar{x}_2.x_3 + - - - -$$ (Yadonsal tek biçim)

Vedensel biçimindeki bir ifadenin her vedenseli ortamdaki tüm değişkenlerin doğal yada değillenmiş biçimlerinin yadalanması ile elde edilmiş ise bu ifadeye vedensel tek biçim denir.

$$f(x_1,x_2) = (x_1+\bar{x}_2)\bullet(\bar{x}_1+\bar{x}_2)$$ (**vedensel** tek biçim)

$(x_1+\bar{x}_2).(\bar{x}_1+x_2).x_2 \rightarrow$ (**vedensel** tek biçim değil. Çünkü $x_2 \rightarrow$ 2 bileşi değil

Herbir lojik fonksiyonun vedensel ve yadonsal tek biçimi vardır

ÖRN/ 2 değişkenli bir ortamda kaç tane yadonsal tek biçim yazılır.

$x_1 . \overline{x_2} + x_1 . x_2$ → yadonsal tek biçim

$\overline{x_1} . x_2 + x_1 . \overline{x_2}$ → " " "

$x_1 . x_2 + x_1 . \overline{x_2} + \overline{x_1} . x_2$ → " " "

x_1	x_2	
0	0	→ $\overline{x_1} . \overline{x_2}$
0	1	→ $\overline{x_1} . x_2$
1	0	→ $x_1 . \overline{x_2}$
1	1	→ $x_1 . x_2$

4 tane

kombinasyon

$$\binom{4}{2} + \binom{4}{3} + \binom{4}{4} = 6 + 4 + 1 = 11$$

Örn: $Z = f(A,B,C) = \bar{A} + B.C$ ifadesi veriliyor. $\bar{Z}$ işleminin yadsısal tek biçimini teoremleri kullanarak bulunuz.

$$\bar{Z} = \overline{\bar{A} + B.C}$$

$$\overset{teo14}{=} \bar{\bar{A}} . \overline{B.C}$$

$$= A.(\bar{B}+\bar{C}) = A\bar{B} + A.\bar{C}$$

$$= A.\bar{B} + A.\bar{C}$$

$$= A.\bar{B}.1 + A.\bar{C}.1$$

$$= A.\bar{B}.(C+\bar{C}) + A.\bar{C}.(B+\bar{B})$$

$$= A.\bar{B}.C + A.\bar{B}.\bar{C} + A.\bar{C}.B + \cancel{A.\bar{C}.\bar{B}}$$

$$= A\bar{B}.C + A\bar{B}\bar{C} + AB\bar{C} \Rightarrow \text{yadsısal tek biçim}$$

RN// Aşağıdaki ifadelerin görünüm biçimlerini bulunuz.

a) $f(A,B,C) = \bar{A}B + A\bar{B} + B\bar{C} \rightarrow$ (Yadalama Biçim)

b) $f(A,B,C,D) = A\bar{B}\bar{C}D + \bar{A}BCD + ABC\bar{D}$ (Yadalama tek biçim)

c) $f(A,B,C) = (A+B+C).(\bar{A}+\bar{B}+C)$ (Vedeleme Tek biçim)

d) $f(A,B,C) = (A+B)CD + \bar{A}\bar{B}(C+D)$

$= ACD + BCD + \bar{A}\bar{B}C + \bar{A}\bar{B}D \rightarrow$ (Yadalama biçim)

RN// Bir sınıfın aydınlatılması 3 anahtar ile kontrol ediliyor. Anahtarların en az ikisi kapalı ise lambalar yanıyor. Diğer durumlarda da lamba yanmıyor.

Bu durumda sınıfın aydınlanmasını sağlayan lojik fonksiyonu,
Aydınlanma fonksiyonunun yadansal tek biçimini,
Aydınlanma fonksiyonunun vedensel tek biçimini:

bulunuz?

Çözüm

$A \rightarrow 1 \rightarrow$ açık

$\bar{A} \rightarrow 0 \rightarrow$ kapalı

A	B	C	f	$\bar{f}$
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

$\boxed{\text{NOT}}$

Yadonsal tek biçimleri tablodan yazdık

(tablodan bulduk)

$f(A,B,C) = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} \Rightarrow$ f nin Yadonsal tek biçimi;

$\bar{f}(A,B,C) = \overline{\bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C}}$

tümleyeni alırsak $\bar{f}$ Vedensel tek biçimi elde edili

$\bar{f}(A,B,C) = (A+B+C).(A+B+\bar{C}).(A+\bar{B}+C).(\bar{A}+B+C) \Rightarrow \bar{f}$ nin Vedensel Tek Biçimi

$\bar{f} = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC \Rightarrow \bar{f}$ nin yadonsal tek biçimi (tablodan bulduk)

$\bar{\bar{f}} = f = \overline{\bar{A}BC + A\bar{B}C + AB\bar{C} + ABC}$

$f = (A+\bar{B}+\bar{C}).(\bar{A}+B+\bar{C}).(\bar{A}+\bar{B}+C).(\bar{A}+\bar{B}+C) \Rightarrow f$ nin Vedensel tek biçimi

Tanım

Değerlerini $S=\{0,1\}$ kümesinden olan değişkenlerin vedelenmiş yada yadalanmış biçimlerinden oluşan ifadeler fonksiyon denir. Fonksiyonların görünümleri 3 türlüdür.

(i) Değişkenlerle gösterim $\Rightarrow$ $X = A\bar{B}C + \bar{A}\bar{B}C + AB\bar{C}$

(ii) 2'lik gösterim $\Rightarrow$ $X = \sum(101, 001, 110)_2$

(iii) 10'luk gösterim $\Rightarrow$ $X = \sum(5,1,6)_{10}$

$X = \sum(1,5,6)_{10}$ (10'luk gösterim)

ÖRN

$Z = f(A,B,C) = \bar{A} + B.C$ fonksiyonunu

i) Teoremler
ii) tablo

kullanarak 3 biçimde ifade ediniz.

Çözüm

i) $Z = \bar{A}.(B+\bar{B}).(C+\bar{C}) + BC(A+\bar{A})$

$\overset{teo4,6}{=} (\bar{A}B + \bar{A}\bar{B}).(C+\bar{C}) + ABC + \bar{A}BC$

$\overset{teo6}{\underset{teo3}{=}} \bar{A}BC + \bar{A}B\bar{C} + \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C} + ABC + \cancel{\bar{A}BC}$

$= \sum(011, 010, 001, 000, 111)_2$ (2 lik gösterim)

- $Z = \sum(3,2,1,0,7)_{10}$

(ii)

A	B	C	$\bar{A}$	B.C	Z	
0	0	0	1	0	1	*
0	0	1	1	0	1	*
0	1	0	1	0	1	*
0	1	1	1	1	1	*
1	0	0	0	0	0	
1	0	1	0	0	0	
1	1	0	0	0	0	
1	1	1	0	1	1	*

$$\Rightarrow Z = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}BC + ABC$$

(tabloda)

$$\bar{Z} = A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C}$$

$$\bar{Z} = \sum(100, 101, 110)_2$$

$$\bar{Z} = \sum(4, 5, 6)_{10}$$

$$\bar{\bar{Z}} = Z = \sum(0, 1, 2, 3, 7)_{10}$$

9/ $\overline{X} = f(A,B,C,D) = ABC + A.\overline{D}$

X fonksiyonunun 2'lik ve 10'luk değişkenlerle olan gösterimini bulunuz.

Çözüm

$\overline{X} = ABC + A\overline{D}$

$\overline{X} = ABC(D+\overline{D}) + A\overline{D}(B+\overline{B})(C+\overline{C})$

$\overline{X} = ABCD + ABC\overline{D} + (AB\overline{D} + A\overline{B}\overline{D})(C+\overline{C})$

$\overline{X} = ABCD + ABC\overline{D} + \cancel{ABC\overline{D}} + A\overline{B}C\overline{D} + AB\overline{C}\overline{D} + A\overline{B}\overline{C}\overline{D}$

$\overline{X} = ABCD + ABC\overline{D} + A\overline{B}C\overline{D} + AB\overline{C}\overline{D} + A\overline{B}\overline{C}\overline{D}$

$\overline{X} = \sum(1111, 1110, 1010, 1100, 1000)_2$

$\overline{X} = \sum(15, 14, 10, 12, 8)$

$\overline{X} = \sum(8, 10, 12, 14, 15)_{10}$

$\overline{X} = \sum(0, 1, 2, 3, 4, 5, 6, 7, 9, 11, 13)_{10} \Rightarrow$ 10'luk gösterim

$= \sum(0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1001, 1011, 1101)_2$

↓ 2'lik gösterim

A	B	C	D	A.B.C	A.$\overline{D}$	$\overline{X}$	
0	0	0	0	0	0	0	
0	0	0	1	0	0	0	
0	0	1	0	0	0	0	
0	0	1	1	0	0	0	
0	1	0	0	0	0	0	
0	1	0	1	0	0	0	
0	1	1	0	0	0	0	
0	1	1	1	0	0	0	
1	0	0	0	0	1	1	→ 8
1	0	0	1	0	0	0	
1	0	1	0	0	1	1	→ 10
1	0	1	1	0	0	0	
1	1	0	0	0	1	1	→ 12
1	1	0	1	0	0	0	
1	1	1	0	0	1	1	→ 14
1	1	1	1	1		1	→ 15

$$\overline{X} = \sum(8,10,12,14,15)_2$$

$$X = \sum(0,1,2,3,4,5,6,7,9,11,13)_{10}$$

Algoritma ve Algoritma Analizi

Bir algoritma, bir problemi çözmek ya da bir işlevi hesaplamak için izlenecek sonlu, açıkça belirtilen talimat dizisidir.

Bir algoritma genel olarak

- *Bir (birkaç) girdi alır.*
- *Sınırlı bir süre içerisinde komutlar bir çıktı üretmektedir.*

Etkili bir talimat, temelde kalem ve kağıt kullanarak gerçekleştirmenin mümkün olduğu kadar basit bir işlemdir.

Algoritmaları İfade Etmek

Algoritmalar şu şekilde gösterilebilir:

- **doğal diller**
  - *ayrıntılı ve belirsizdir.*
  - *nadiren karmaşık veya teknik algoritmalar için kullanılır.*
- **Pseudocode(sözde kod), akış diyagramları:**

  --- algoritmaları ifade etmek için yapısal yöntemlerdir.

  --- doğal dilde ifadelerde belirsizliklerden kaçınır.

  --- belirli bir uygulama dilinden bağımsızdır.
- **Programlama dilleri:**

  --- algoritmaları bir bilgisayar tarafından yürütülebilecek biçimde ifade etmeyi amaçlar.

  --- algoritmaları belgelemek için kullanılabilir.

Örnek:

Problem: Sıralanmamış bir listede en büyük elemanı bulmak

Fikir: Her elemana bir kere bakmak.

Doğal Dil:

- Listedeki ilk elemanın en büyük olduğunu varsay.
- Listenin sonuna kadar daha büyük bir sayı var mı diye arat.
- Liste tarama işlemi bittiğinde en son not edilen en büyük elemandır.

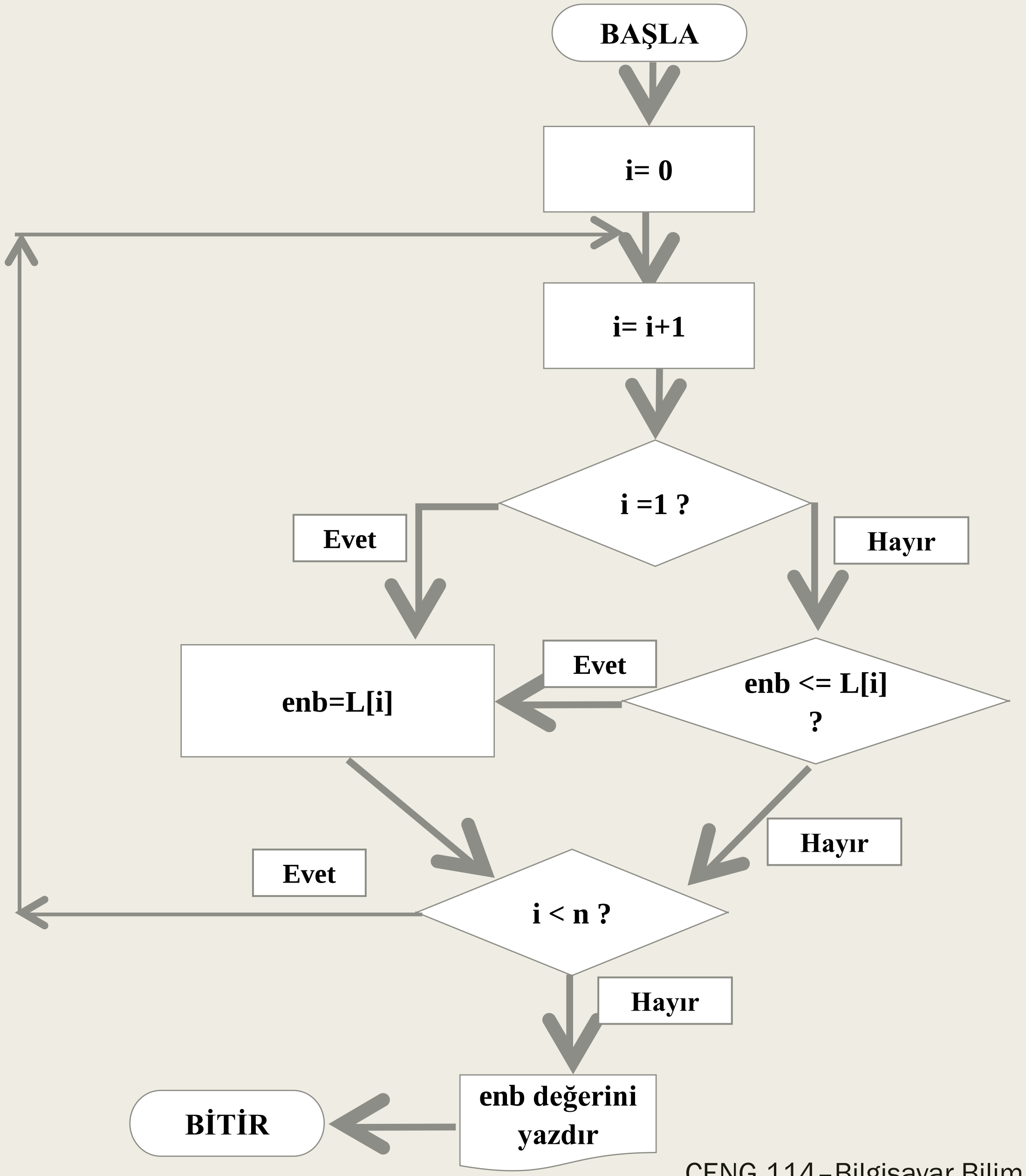
Örnek:

Sözde Kod:

Algorithm LargestNumber
    Input: A non-empty list of numbers L.
    Output: The *largest* number in the list L.
    largest $\leftarrow L_0$
    for each *item* **in** the list $L_{i \geq 1}$, **do**
        if the *item* > *largest*, **then**
            largest $\leftarrow$ the *item*
    return *largest*

Örnek:

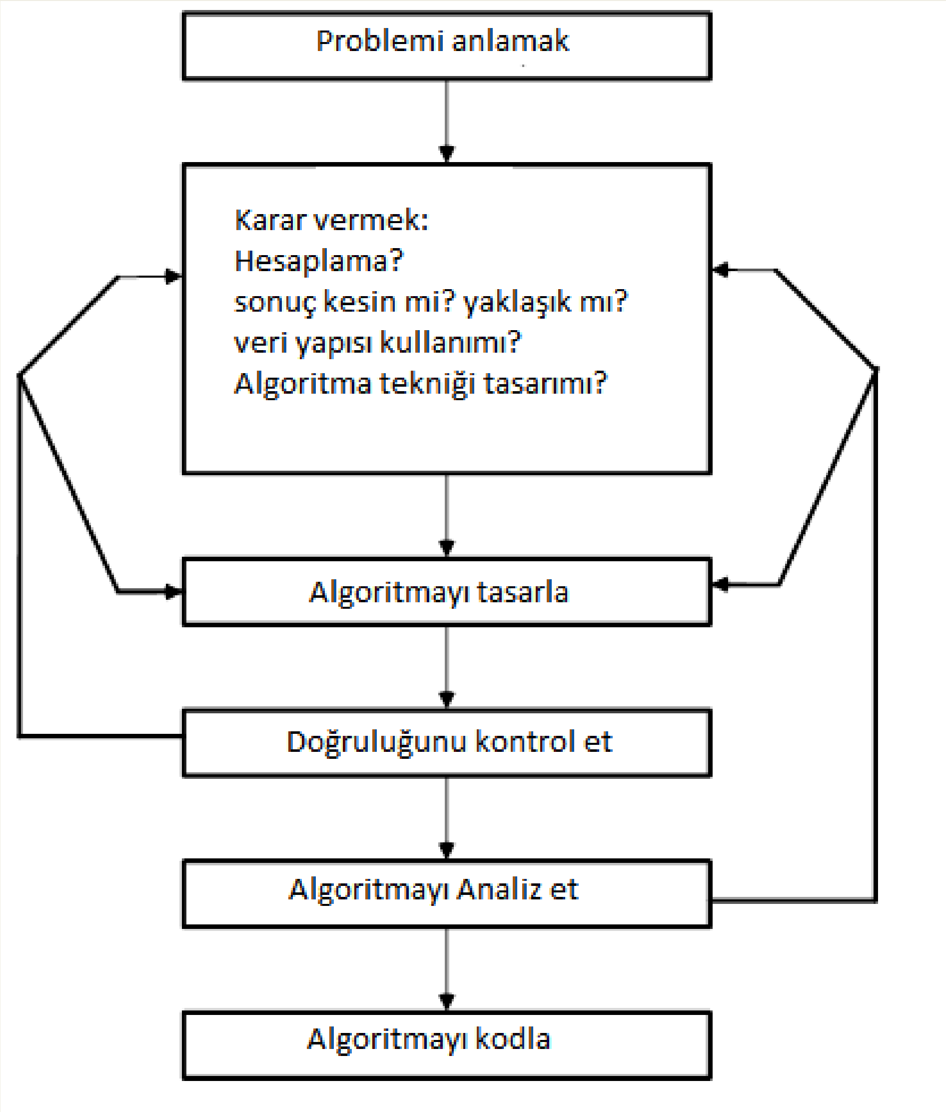
Akış Diyagramı



Algoritmanın Özellikleri

- **Effectiveness (Etkinlik)**
  - *Talimatlar basit olmalı.*
  - *kalem ve kağıtla yapılabilir.*
- **Definiteness (Kesinlik)**
  - *Talimatlar net*
  - *Anlamı tek olmalıdır.*
- **Correctness (Doğruluk)**
  - *Algoritma doğru cevabı verir (Olası tüm durumlar için)*
- **Finiteness (Sonluluk)**
  - *Algoritma makul sürede durmalı ve bir çıktı üretmelidir.*

Algoritma Tasarım Süreci



Algoritmaların Analizi

- Bir algoritmanın complexity'sini (karmaşıklığı) çalışma
  - *Algoritma ne kadar iyi?*
  - *Diğer algoritmalarla karşılaştırma işlemi nasıl yapılacak?*
  - *En iyi yazılabilecek algoritma bu mudur?*

- Karmaşıklık
  - *Alan Karmaşıklığı*
    - Bit sayısı
    - Eleman sayısı

  - *Zaman Karmaşıklığı*
    - Toplamda çalıştırılacak işlem sayısı
      - *Modele göre değişir*
      - *RAM*

Algoritmaların Run-Time (Çalışma Zamanı) Analizi

- Algoritma karmaşıklığı, problemin boyutunu gösteren parametre n' nin bir fonksiyonu olarak hesaplanabilmektedir.

- Zaman karmaşıklığı, T (n), algoritmanın en önemli işlemi olan - temel işlem olarak adlandırılan – işlemin çalıştırılma sayısı olarak hesaplanabilir.

- Space (Alan) karmaşıklığı, S (n), genellikle algoritmanın yürütülmesi sırasında kullanılan bellek alanının büyüklüğü olarak hesaplanır.

Tablo Metodu

- *Tablo Metodu, bir algoritmanın karmaşıklığını hesaplamak için kullanılır*

Örnek: Bir dizinin elemanlarını toplama

	İşlem sayısı
`top = 0`	1
`for (i=0; i<n; i++){`	1+n+1+n
`top = top + a[i]}`	n
`print top`	1
	3n+4

Kaç işlem yapılır? T(n)=3n+4 (Yürütme zamanı)

Tablo Metodu

- Örnek:

<u>Matris Toplama</u>

a, b, c ‘nin mxn boyutunda matrisler olduğunu varsayalım.

	işlem	toplam
for (i=0; i<m; i++){	1+m+1+m	2m+2
for (j=0; j<n; j++){	m(1+n+1+n)	2mn+2m
c[i,j] = a[i,j] + b[i,j] } }	mn	mn
		3mn+4m+2

Eğer m=n ise $T(n) = 3n^2 + 4n + 2$ (Yürütme zamanı)

Asimptotik Notasyon Ve Temel Verimlilik Sınıfları

(Büyüme Sırası) Order of growth

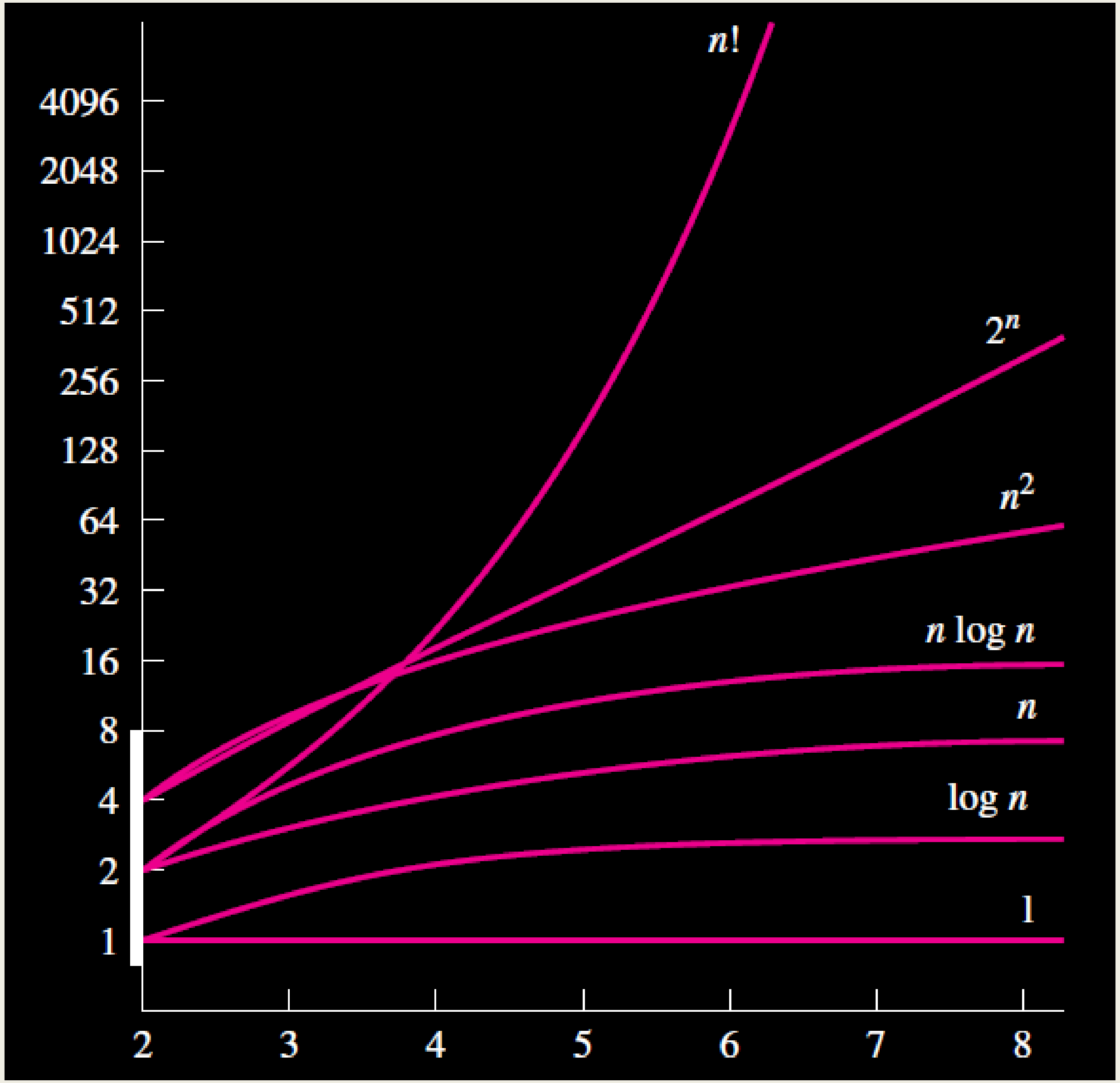
- En önemlisi : $n \rightarrow \infty$'a giderken algoritmanın performansı hangi sınırlarda bunu anlayabilmektir.

- Örnek:
  - *İki katı kadar hızlı bir bilgisayarda algoritma ne kadar hızlanıyor?*
  - *Girdi boyutu iki katına çıktığında algoritma ne kadar yavaşlıyor?*

$n \to \infty$ giderken bazı önemli fonksiyonların değerleri

Tablo: Algoritma analizi için bazı önemli fonksiyonların değerleri

n	$\log_2 n$	n	$n \log_2 n$	n^2	n^3	2^n	$n!$
10	3.3	10^1	$3.3 \cdot 10^1$	10^2	10^3	10^3	$3.6 \cdot 10^6$
10^2	6.6	10^2	$6.6 \cdot 10^2$	10^4	10^6	$1.3 \cdot 10^{30}$	$9.3 \cdot 10^{157}$
10^3	10	10^3	$1.0 \cdot 10^4$	10^6	10^9		
10^4	13	10^4	$1.3 \cdot 10^5$	10^8	10^{12}		
10^5	17	10^5	$1.7 \cdot 10^6$	10^{10}	10^{15}		
10^6	20	10^6	$2.0 \cdot 10^7$	10^{12}	10^{18}		


n!
2^n
n^2
n log n
n
log n
1
4096
2048
1024
512
256
128
64
32
16
8
4
2
1
2
3
4
5
6
7
8

Örnek:

- 2^100'ü hesaplamak saniyede 10^{12} işlem yapan bir bilgisayar için $4x10^{10}$ yıl alacaktır.
- Bu, 100! Değerini hesaplamak için gereken süreden kısadır. Fakat, 100!i hesaplamak ise dünya gezegeninin tahmini yaşından 4,5 milyar ($4.5\ .10^{9}$) yıl daha uzundur.
- 2^n ve n! fonksiyonlarının büyüme sıraları arasında muazzam bir fark vardır.

Asimptotik Büyüme Dereceleri

Asimtotik Analiz

$n \to \infty$ iken $T(n)$'nin büyümesi nedir?

Asimtotik Notasyon

O	Ω	Θ
Büyük O	Büyük omega	Teta
(Big O)	(Big Omega)	(Theta)

O Notasyonu (Üst Sınır)

Ω Notasyonu (Alt Sınır)

Θ Notasyonu (Sıkı Sınır, Ortalama durum)

Kısaltmalar ve Anlamları

O	Ω	Θ
$\leq$	$\geq$	$=$

$\Theta(n^2)$ algoritması aynı problemi çözen bir $\Theta(n^3)$ algoritmasından daha hızlıdır.

Asimptotik Büyüme Dereceleri

Fonksiyonların büyüme hızlarını karşılaştırmak için kullanılan, sabit çarpanları ve küçük girdi boyutlarını yok sayan bir yöntem.

- $O(g(n))$: $g(n)$ fonksiyonundan daha hızlı büyümeyen $f(n)$ fonksiyonlarını kapsar.
- $\Theta(g(n))$: $g(n)$ fonksiyonları ile aynı derecede büyüyen $f(n)$ fonksiyonlarını gösterir.
- $\Omega(g(n))$: en az $g(n)$ fonksiyonları kadar hızda büyüyen $f(n)$ fonksiyonlarını belirtmek için kullanılır.

Big-oh

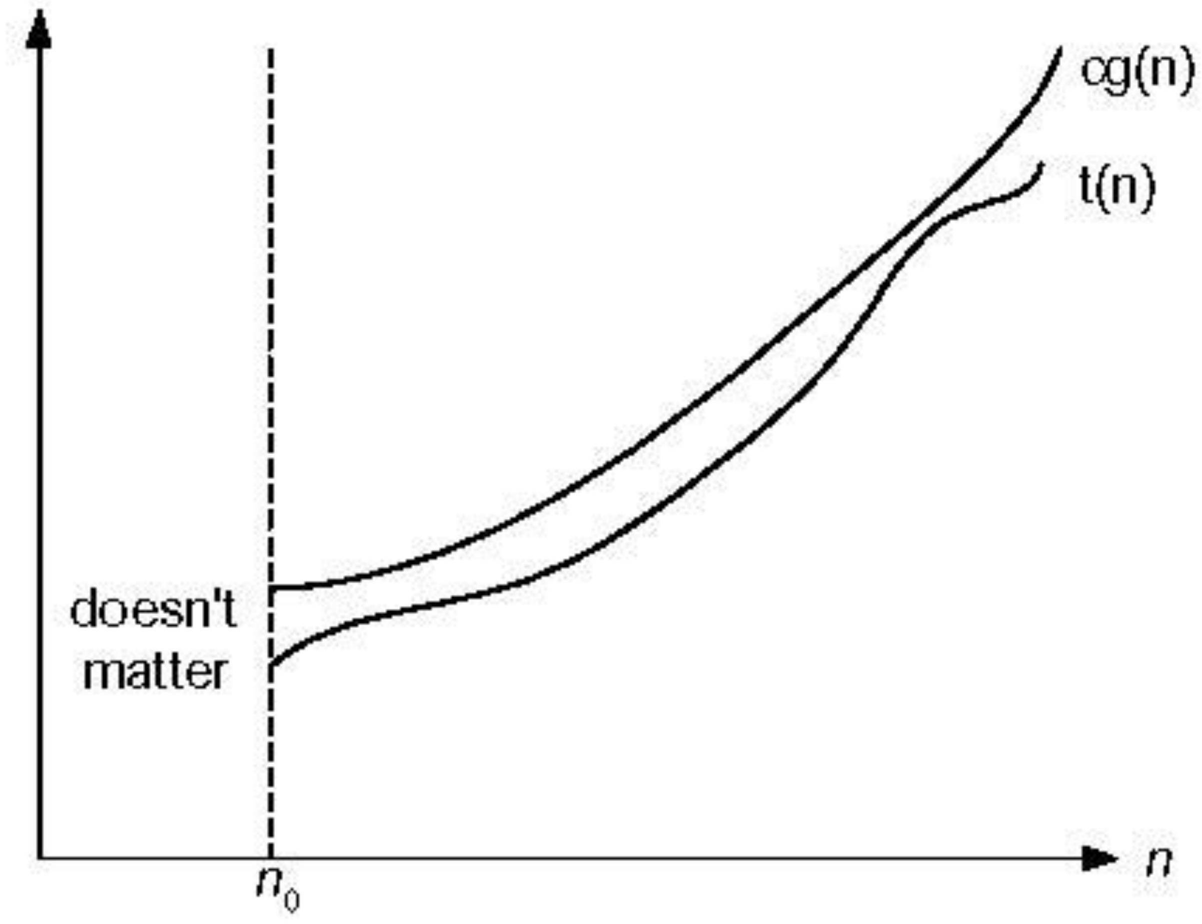


Figure 2.1 Big-oh notation: $t(n) \in O(g(n))$

Big-omega

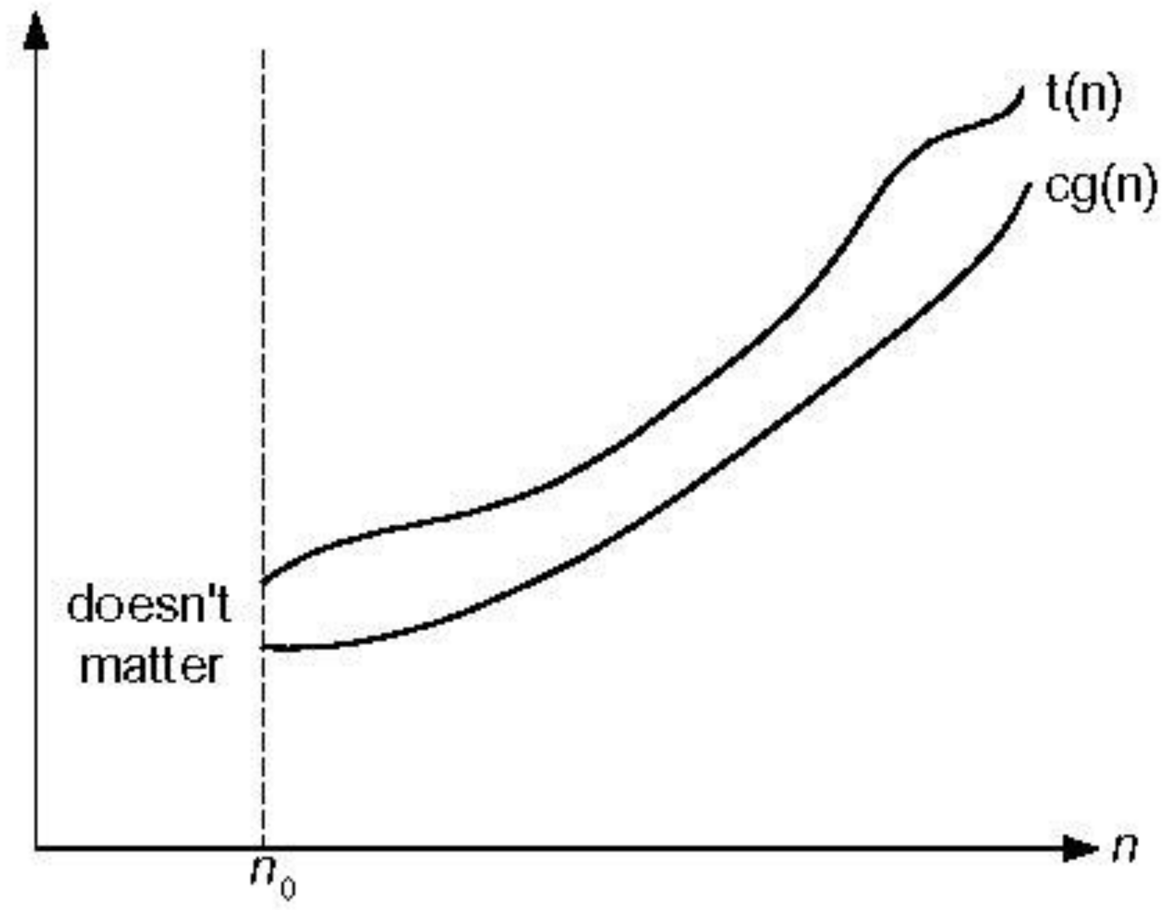


Fig. 2.2 Big-omega notation: $t(n) \in \Omega(g(n))$

Big-theta

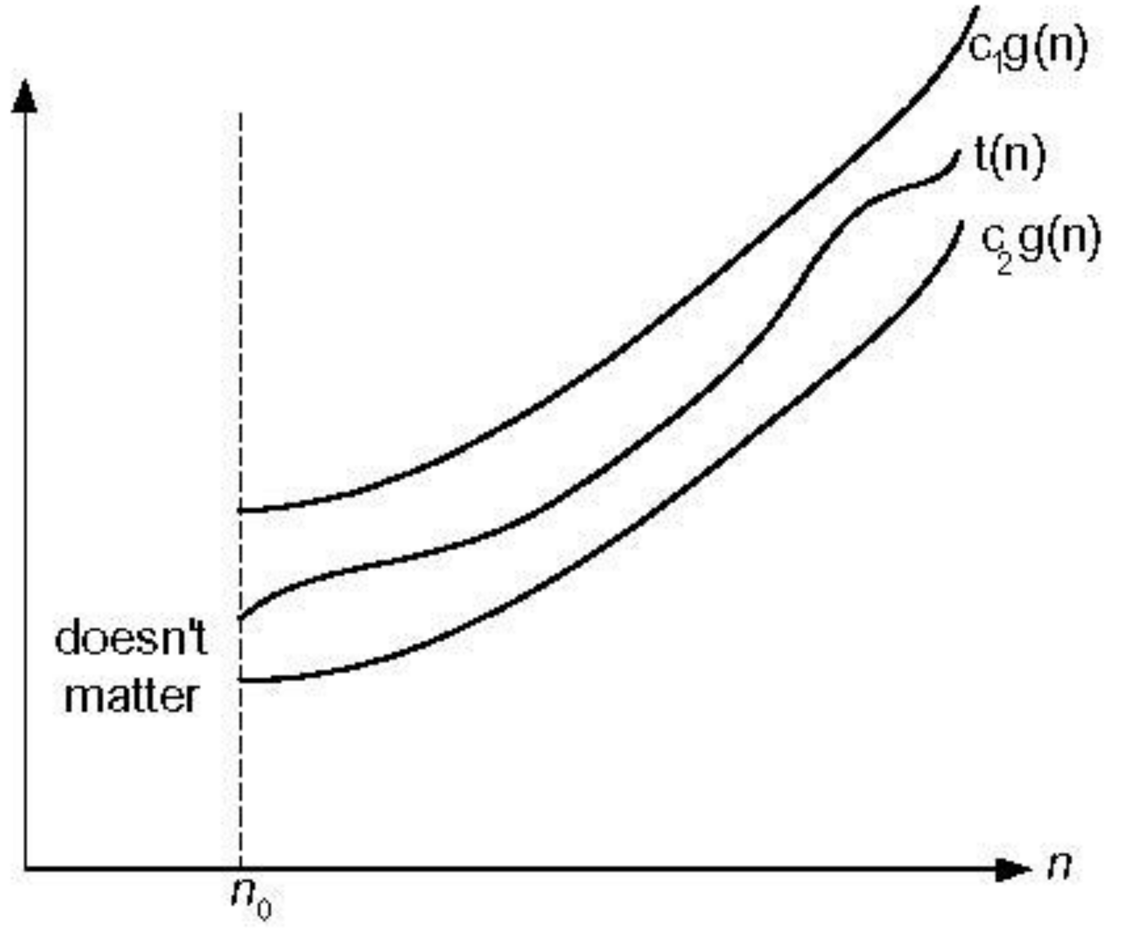
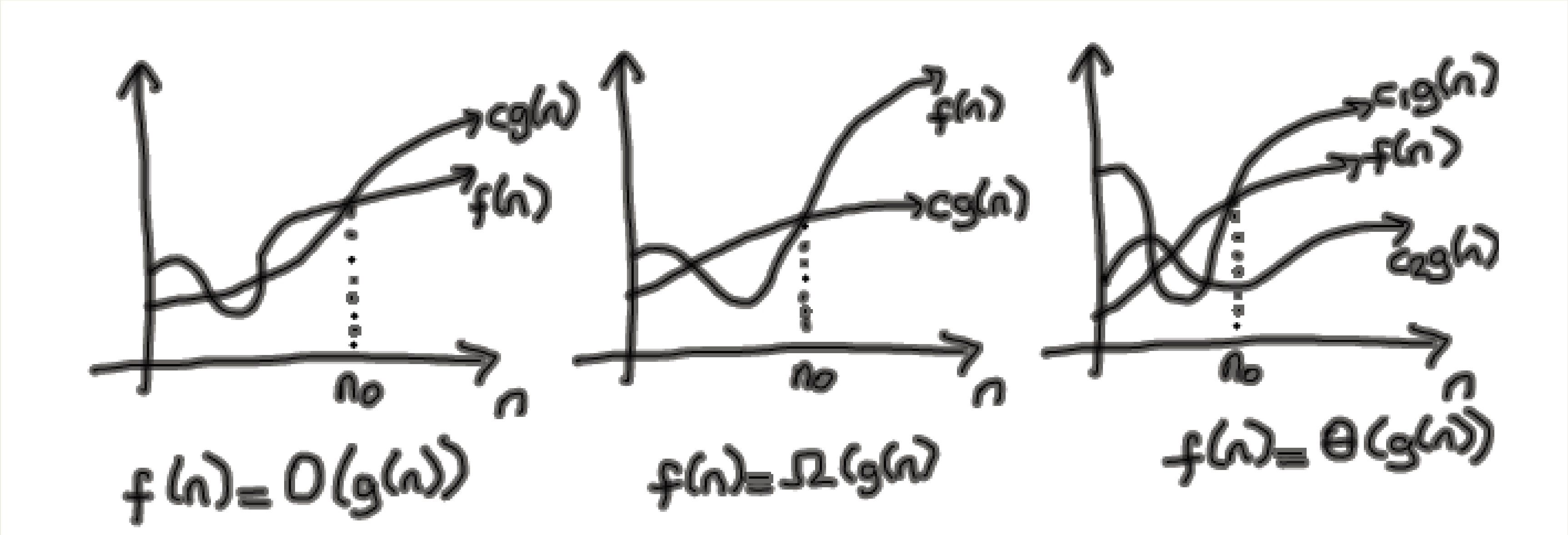


Figure 2.3 Big-theta notation: $t(n) \in \Theta(g(n))$

Asimtotik Notasyonların Grafik Üzerinde Gösterimi



*) Asimtotik analiz $n \to \infty$ iken fonksiyonun büyümesi ile ilgilidir.

Big *O* Formal Tanımı

Tanım: $f(n) \in O(g(n))$ ise, f(n) fonksiyonunun büyüme derecesi, $g(n)$'in büyüme sabit bir sayı ile çarpımının büyüme derecesinden küçüktür.

$$f(n) \leq c\, g(n)\ ,\ \ \forall\ \ n \geq n_0$$

Eşitsizliğini sağlayan pozitif bir sabit c ve pozitif bir tamsayı n_0 vardır.

--- O notasyonu ilk olarak alman Matematikçi Bochmann tarafından 1894 yılında tanımlanmıştır.

--- Algoritmanın en kötü durum analizini yapmak için kullanılan notasyondur.

Örnek:

$T(n)=3n+8 = O(n)$ dir. ($3n+8 \in O(n)$)

$3n+8 <= c*n$

$n>=1$ için $3n+8<=3n+8n<=11n$ ($c=11$, $n_0=1$)

$n>=4$ için $3n+8<= 5n$ ($c=5$, $n_0=4$)

Diğer c ve n_0 değerleri bulunabilir.

Örnek:

$T(n)=3n+8= O(n^2)$ dir. ($3n+8 \in O(n^2)$)

$3n+8<= c*n^2$

$n>=5$ için $3n+8<= c * n^2$ ($c=1$, $n_0=5$)

$n>=3$ için $3n+8<= c * n^2$ ($c=2$, $n_0=3$)

Diğer c ve n_0 değerleri bulunabilir.

Örnek:

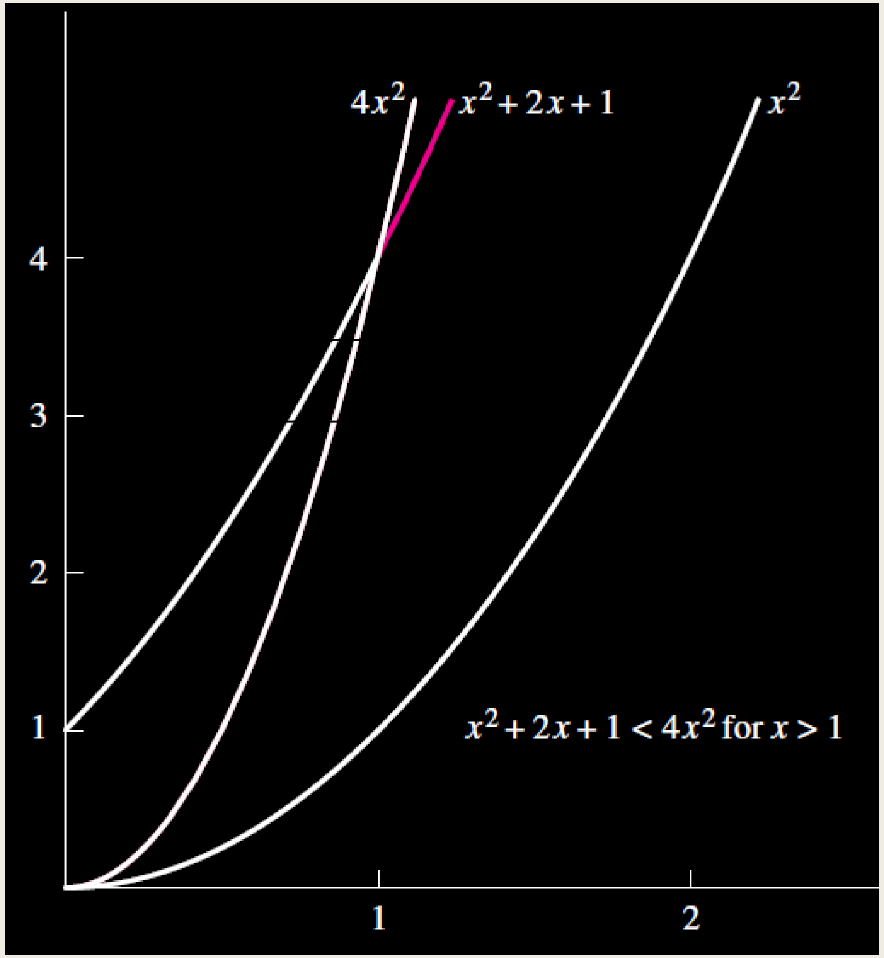
- $f(x) = x^2 + 2x + 1$ ise $O(x^2)$ dir.
- Eğer $x>1$ ise $x<x^2$ dir ve, eğer $x>1$ ise $1<x^2$ dir.
- Çünkü:

$$0 \leq x^2 + 2x + 1 \leq x^2 + 2x^2 + x^2 = 4x^2$$

$x>1$ olduğu her değer için

- Böylece, $c = 4$ ve $n_0 = 1$ **alınırsa** $f(x) \in O(x^2)$ elde edilir.

Grafiği



Örnek:

A_{nxm} ve B_{mxr} tipi 2 matrisi çarpıp C_{nxr} isimli matrise sonuçları yazan bir alg. yazınız ve zaman karmaşıklığını hesaplayınız.

```
procedure matmul ( int array:A, B, C);
        A[n][m], B[m][r], C[n][r]

begin
    x,y,z : integer;

    for x:=0 to n do ------ 1+n+1+n
    begin
        for z:=0 to r do  ------>  (1+r+1+r)·n
        begin
            aratoplam:=0;  ------>  n·r
                                                  (2m+2)·n·r
            for y:=0 to m do  ------>
                                                          -> m·n·r
            begin
                aratoplam:= A[x][y] * B[y][z] + aratoplam;
            end;

            C[x][z]:=aratoplam  -> n·r

        end;
    end;
end;
```

$$T(n,m,r) = 2n+2+(2r+2).n+n.r+(2m+2).r.n+n.m.r+rn$$

$$= 3mnr + 6nr + 4n + 2$$

$m=n=r$ olursa

$$T(n) = 3n^3 + 6n^2 + 4n + 2$$

$$\boxed{O(n^3)} \rightarrow \text{karmaşıklık!!!}$$

Önemli Teoremler

Teo: Eğer $f(n) = O(g(n)) \Rightarrow c.f(n) = O(g(n))$

İspat)

$f(n) = O\,g(n) \Rightarrow \exists k, m \quad \forall n \geq m, \; f(n) \leq |c|.g(n)$

$\Rightarrow |c.f(n)| \leq k.|c|.|f(n)| \Rightarrow c.f(n) = O(g(n))$

Teo: Eğer $f(n) = O(g(n))$ ve $h(n) = O(g(n)) \Longrightarrow$

$(f+h)(n) = O(g(n))$ 'dir.

İspat:

$f(n) = O(g(n)) \Rightarrow \exists k, m_1 : \forall n > m_1 \quad |f(n)| \leq k \cdot |g(n)|$ ---(1)

$h(n) = O(g(n)) \Rightarrow \exists \ell, m_2 : \forall n > m_2 \quad h(n) \leq \ell \cdot |g(n)|$ ---(2)

$$m = \max\{m_1, m_2\} \overset{(1),(2)}{\Longrightarrow} \forall n > m_1$$

$$|f(n) + h(n)| \leq |f(n)| + |g(n)| \leq k \cdot |g(n)| + \ell |g(n)|$$

$$= (k+\ell) \; |g(n)|$$

$$\Rightarrow (f+h)(n) = O(g(n))$$

Teorem!)

Eğer $f(n) = O(g(n))$, $h(n) = O(e(n)) \Rightarrow (f \cdot h)(n) = O((g.e)(n))$

İspat!)

$f(n) = O(g(n)) \Rightarrow \exists k, m_1 : \forall n \geq m_1, |f(n)| \leq k.|g(n)|$

$h(n) = O(e(n)) \Rightarrow \exists \ell, m_2 : \forall n \geq m_2, |h(n)| \leq \ell\, |e(n)|$

$m = \max\{m_1, m_2\} \Rightarrow \forall n \geq m$ için;

$|f(n).h(n)| \leq |f(n)| . |h(n)| \leq k.|g(n)| . \ell\, |e(n)| =$

$= k.\ell.|g(n).e(n)|$

$\Rightarrow (f.h)(n) = O((g.e)(n))$

Çalışma Sorusu:

Teorem:

Eğer $p(n) = a_k n^k + a_{k-1} n^{k-1} + \dots + a_2 n^2 + a_1 n + a_0$

$\Rightarrow$ $p(n) = O(n^k)$'dır.

Kanıt??

Bazı Önemli Fonksiyonların Büyüme Dereceleri

- Tüm logaritmik fonksiyonlar $\log_a n$ aynı asimptotik sınıfa sahiptir.

  $O(\log n)$ logaritmanın tabanı $a > 1$ önemli değil.

- Aynı derece *k'ye sahip olan tüm polinomlar* aynı asimptotik sınıfa sahiptir. :
- $a_k n^k + a_{k-1} n^{k-1} + \ldots + a_0 \in O(n^k)$
- Üstel fonksiyonlar a^n, a değerine göre farklı büyüme sınıfına aittir.
- order $\log n$ < order n^α $(\alpha>0)$ < order a^n < order $n!$ < order n^n

Temel Asimptotik Verimlilik Sınıfları

1	**sabit**
log *n*	**logaritmik**
n	**lineer**
n* log *n	***n*-log-*n* or linerritmetik**
n^2	**quadratic**
n^3	**kübik**
2^n	**üstel**
***n*!**	**faktoriyel**

Ω- Formal Tanımı

- **Tanım:** $f(n) \in \Omega(g(n))$ ise, f(n) fonksiyonunun büyüme derecesi, $g(n)$'in sabit bir sayı ile çarpımının büyüme derecesinden büyük veya eşittir.

$$f(n) \geq c\, g(n)\ ,\ \ \forall\ \ n \geq n_0$$

Eşitsizliğini sağlayan pozitif bir sabit c ve pozitif bir tamsayı n_0 vardır.

Örn: T(n)=3n+5 $\in \Omega(n)$

3n+5 >= 3n, tüm n>=1 için sağlanır (c=3, n_0=1)

Örnek: $\underbrace{n}_{f(n)} = \Omega(\underbrace{\lg n}_{g(n)})$ 'dir.

$$\boxed{\log_2^n = \lg n}$$

$c.g(n) \le f(n)$

$1.\lg n \le n$

$n=2$ için $1 \le 2$

$c=1$ ve $\forall n \ge n_0$ için, $c.gn \le f(n)$

olduğundan $f(n) = \Omega(g(n))$ 'dir.

$\boxed{n = \Omega(\lg n) \text{ 'dir.}}$

Çalışma Sorusu: $n^3 = \Omega(n^2)$?

Θ Formal Tanımı

- **Tanım:** $f(n) \in \Theta(g(n))$ ise, f(n) fonksiyonunun büyüme derecesi, $g(n)$ fonksiyonun bir sabit katından yüksek aynı zamanda $g(n)$ fonksiyonun bir sabit katından da düşük olmaktadır.

$$c_1 g(n) \leq f(n) \leq c_2 g(n), \quad \forall \; n \geq n_0$$

Eşitsizliğini sağlayan pozitif sabit c_1, c_2 *sayıları* ve pozitif bir tamsayı n_0 vardır.

$\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$ 'dir.

Hem alttan, hem de üstten sınırlıdır.

$5n^2 + n - 7 = \Theta(n^2)$ 'dir

*)

Θ notasyonunda küçük terimler ihmal edilir

Örnek: $\frac{1}{2}n^2 - 2n \in \Theta(n^2)$ olduğunu gösteriniz.

$f(n) = \frac{1}{2}n^2 - 2n \quad g(n) = n^2$ dir.

$c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n) \quad \forall n \geq n_0$

$c_1, c_2 > 0$

$c_1 < c_2$

$\frac{1}{4} \cdot n^2 \leq \frac{1}{2}n^2 - 2n \leq \frac{1}{2} \cdot n^2 \Rightarrow \forall n$ için doğrudur.

$\frac{1}{4} n_0^2 = \frac{1}{2} n_0^2 - 2n_0$

$2n_0 = \frac{1}{4} n_0^2$

$\boxed{n_0 = 8} \Rightarrow$ eşik değeri.

Böylece: $c_2 = \frac{1}{2} > 0$ $\quad n_0 = 8$ için

$c_1 = \frac{1}{4} > 0$

$\frac{1}{2} n^2 - 2n \in \Theta(n^2)$ dir.

$\Theta(n)$ $\quad n_0 = 9$ için

$$\frac{1}{4} \cdot 9^2 \leq \frac{1}{2} \cdot 9^2 - 2 \cdot 9 \leq \frac{1}{2} \cdot 9^2$$

$$\frac{81}{4} \leq \frac{45}{2} \leq \frac{81}{2}$$

Kaynaklar

- *Discrete Mathematics and Its Applications*, Kennet H. Rosen

(Ayrık Matematik ve Uygulamaları, Kennet H. Rosen (Türkçe çeviri), Palme yayıncılık)

- *Discrete Mathematics: Elementary and Beyond*, L. Lovász, J. Pelikán, K. Vesztergombi, 2003.

- *Introduction to Algorithms*, T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, 2009.

- *Introduction To Design And Analysis Of Algorithms*, A. Levitin, 2008.