ALGORITM TAHLILI.

1. Algoritm ishlash vaqti.
2. Algoritmlarni taqqoslash.
3. Algoritmni ishlash tezligi.
4. Tahlil turlari.
5. Asimtotik baholash.
6. Dastur uchun hisoblash vaqtini aniqlash.

Algoritmlarning samaradorligi

- **Algoritm samaradorligi va uni o'lchash**
- **Muammoning hajmiga qarab algoritmning vaqt murakkabligi**
- **Eng yomon, o'rtacha va eng yaxshi holatlar**
- **Nimani tezlashtirish kerak: kompyutermi yoki algoritmmi?**

Algoritmlarning samaradorligi

- Ko'pincha bir xil muammoni hal qilish uchun turli xil algoritmlardan foydalanish mumkin.
- Algoritmni qanday tanlash mumkin?
- Dasturlash ikkita (ko'pincha bir-biriga zid) maqsadga ega

- Tushunish, kodlash va disk raskadrovka qilish oson bo'lgan algoritmni ishlab chiqing

  "Dasturiy ta'minot muhandisligi" fanini o'rganish mavzusi

- Kompyuter resurslaridan samarali foydalanadigan algoritmni ishlab chiqish

  "Ma'lumotlar tuzilmalari va algoritmlarni tahlil qilish" fanini o'rganish mavzusi.

Algoritmlarning samaradorligi

- **Asosiy resurslar**
  - **Algoritmning bajarilishi vaqti**
    - **Muammoni hal qilish uchun zarur bo'lgan soda qadamlar soni bilan belgilanadi**
  - **Algoritm tomonidan ishlatiladigan xotira maydoni**
    - **Ma’lumotni saqlashdagi operativ xotira yoki xotira hajmi bilan aniqlanadi**

Qo’shimcha sifatida:

- **Algoritmni kodlashning murakkabligi**

Dasturchining algoritmni kodlash va tuzatishga sarflagan vaqti bilan belgilanadi

Algoritmlarning samaradorligini qanday o'lchash mumkin?

- **Mumkin bo'lgan usullar**

1. **Algoritmlarni tajribaviy (empirik) taqqoslash**

Dasturlarni bevosita ishga tushirishda vaqt (xotira) qiymatlarini solishtirish

1. **Algoritmlarning asimptotik tahlili**

Turli omillarga bog'liq holda vaqt (xotira) qiymatlarining nazariy hisoblash usulini ishlab chiqish

Amalga oshirish vaqtini nima belgilaydi?

- **Mashinani yuklashdan**
- **Operatsion tizimdan**
- **Kompilyatordan**
- **Kirish ma'lumotlari qiymatlarining o'ziga xos xususiyatlaridan**
- **Vazifa hajmidan**

Bajarish vaqti T ning vazifaning n o'lchamiga bog'liqligi T(n) funktsiyasi bilan ifodalanadi.

Vaqtning o'lchamga bog'liqligi

- **Misol 1. Maksimalni topish**

```
int largest(int array[], int n) {
  int currlarge = 0;
  for (int i=1; i<n; i++)
    if (array[currlarge] < array[i])
      currlarge = i;
  return currlarge;
}
```

$$T(n) = c_1 n + c_2$$

- **2-misol. Summani hisoblash**

```
sum = 0;
for (i=1; i<=n; i++)
   for (j=1; i<n; j++)
     sum++;
```

$$T(n) = c_1 n^2 + c_2$$

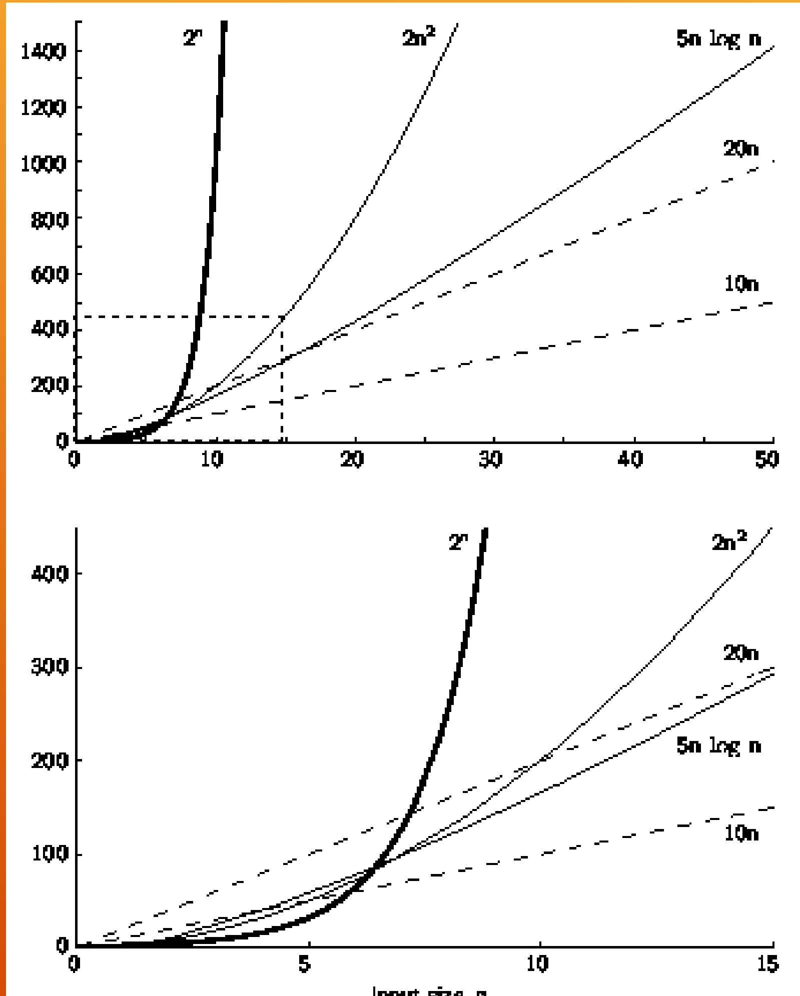
- **3-misol: O’zlashtirish**

```
count = 0;
```

$$T(n) = c$$

Funktsiyalarning o'sishining xarakteristikasi

- **T(n) funksiyaning shakliga qarab, uning o'sish xarakteri har xil bo'lishi mumkin**



Eng yaxshi, eng yomon va o'rtacha holatlar

- **Xuddi shu kirish ma'lumotlarining o'lchami n uchun algoritmning bajarilish vaqti har xil bo'lishi mumkin**

- Misol. n oʻlchamli massivdan K elementni ketma-ket qidirish

Massiv elementlari, birinchisidan boshlab, K topilgunga qadar birma-bir tekshiriladi.

***Eng yaxshi: 1-pozitsiyada K* topilsa**

***Eng yomon: n*-pozitsiyada K topilsa**

O’rtacha holat: **o'rtacha K (n+1)/2 taqqoslashdan keyin topilsa**

Qaysi holatlarni baholash kerak?

- **Algoritmning harakatini o'rtacha tahlil qilish eng oqilona, ammo eng qiyin**

  Qiymatlarning taqsimlanishini bilishni talab qiladi (ma'lum ma'lumotlarning paydo bo'lish chastotasini bilish)

- **Algoritmning eng yomon holatini real vaqt algoritmlarida tahlil qilish muhim**

  Misol - transport dispetcherlik tizimlari

Kompyuterni tezlashtirasizmi yoki algoritmni?

- **Agar kompyuterdan 10 baravar samarali foydalansangiz nima bo'ladi?**

T(n)	n	n'	O'zgartirish	n'/n
$10n$	1,000	10,000	$n' = 10n$	10
$20n$	500	5,000	$n' = 10n$	10
$5n \log n$	250	1,842	$\sqrt{10}\, n < n' < 10n$	7.37
$2n^2$	70	223	$n' = \sqrt{10n}$	3.16
2^n	13	16	$n' = n + 3$	---

Kompyuterni tezlashtirasizmi yoki algoritmmi?

- **Turli xil murakkablikdagi algoritmlarning mutlaq vaqt sarfi**

T(*n*)	n=10	$n=10^3$	$n=10^6$
log *n*	**0.2 сек**	**0.6 сек**	**1.2 сек**
n	**0.6 сек**	**1 час**	**16.6 час**
n^2	**6 сек**	**16.6 час**	**1902 года**
2^n	**1 час**	**10^{295} лет**	**10^{300000} лет**

Algoritmlarning asimptotik tahlili

- **Asimptotik tahlilning maqsadlari**
- **O-ramzlari**
- **Algoritmlarning asimptotik tahliliga misollar**
- **Muammolarning asimptotik murakkabligi**
- **Vaqt va fazo murakkabligi**

Asimptotik tahlil

- **Algoritmlarni eksperimental taqqoslash mashaqqatli**
- **Amalda ko'pincha algoritmlarning oddiyroq asimptotik tahlili qo'llaniladi**
- **Algoritmlarning asimptotik tahlili katta qiymatli n uchun algoritmlarning murakkabligi (T(n) ko‘rinishdagi) nazariy baholarini olish va solishtirishga qaratilgan.**

Asimptotik tahlil

- **Algoritmlarning asimptotik tahlilida matematik asimptotik analizda qabul qilingan belgi qo'llaniladi.**

- ***O*-ramzlari**
  - $o(f(n))$ **kichiklik tartibini baholash**
  - $O(f(n))$ **yuqori chegara bahosi**
  - $\Omega(f(n))$ **pastki chegara bahosi**
  - $\Theta(f(n))$ **yuqori va pastki chegaralarni baholash**

Asimptotik tahlil : O()

- **Ta'rif**

  Har qanday $n > n_0$ uchun $f(n) \leq cg(n)$ bo‘ladigan $c>0$ va $n_0>0$ konstanta qiymatlar mavjud bo‘lsa, manfiy bo‘lmagan f(n) funksiya O(g(n)) deyiladi.

- **Misol**

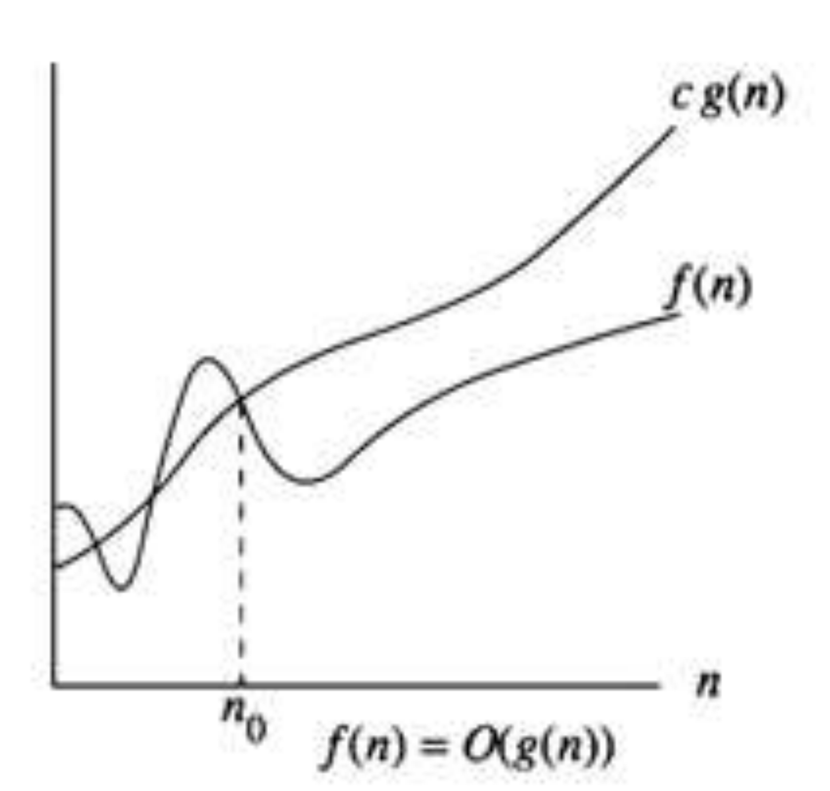
  Algoritmning vaqt murakkabligi [eng yaxshi, o'rtacha, eng yomon] holatda $O(n^2)$ dir.

- **Ma'nosi**

  Barcha yetarli darajadagi katta kirishlar uchun ($n>n_0$), algoritm [eng yaxshi, o'rtacha, eng yomon] holatda har doim cg(n) dan kamroq bosqichlarda ishlaydi.

Asimptotik tahlil: O()

f(n)=O(g(n)) ifoda f(n) ning doimiy koeffitsientgacha g(n) funksiyasidan tez o‘smaydigan funksiyalar sinfiga tegishli ekanligini bildiradi.



Misol. Agar $T(n) = 3n^2$ bo‘lsa, T(n) ya’ni $O(n^2)$ bo‘ladi.

Asimptotik tahlil: O()

- **O() yuqori chegarani belgilaydi**

- **Yuqori chegarani tanlashda, mumkin bo'lgan eng kichik qiymatdir**

- **Misol.**

  T(n) = 3n^2 va O(n^3) bo‘lsa-da, biz ko‘proq ma’lumot beruvchi variant sifatida O(n2) ni tanlaymiz.

Asimptotik tahlil: O()

- Misol 1. **Massivda chiziqli qidiruv (o'rtacha holat)**

  T$(n) = c_s n/2$

  Barcha $n > 1$, $c_s n/2 \leq c_s n$

  Shunday qilib, ta'rifga ko'ra,

  T(n) **va** O(n) **при** $n_0 = 1$ **и** $c = c_s$

Asimptotik tahlil: O()

- Misol 2. **O'rta holatda T(n) = $c_1n^2 + c_2n$ bo'lsin**

  $c_1n^2 + c_2n \;\le\; c_1n^2 + c_2n^2 \;\le\; (c_1 + c_2)n^2$

  barcha $n > 1$

  T(n) $\le cn^2$ **bunda** $c = c_1 + c_2$ **va** $n_0 = 1$**.**

  U holda ta’rif bo’yicha T(n) **ning bahosi O(n^2) bo’ladi**

- Misol 3. T(n) = c. **Ushbu holda:** T(n) **ning bahosi O(1) bo’ladi.**

Asimptotik tahlil: O()

- **Umumiy xato**

“Mening algoritmim uchun eng yaxshi holat n=1, chunki algoritm eng tezkor holatda” – LEKIN BU TO'G'RI EMAS!

- **O() o'sish xarakteristikasini tavsiflaydi, chunki n ∞ ga intiladi**
- **Eng yaxshi holat - n o'lchamdagi kirish ma'lumotlarini qayta ishlash bir xil o'lchamdagi boshqa ma'lumotlarga nisbatan eng kam vaqt talab qiladigan holatda.**

- **Umumiy xato**

Ko'pincha eng yomon holat yuqori chegara bilan chalkashib ketadi.

- **Yuqori chegara o'sish xarakterini tavsiflaydi, chunki n ∞ ga intiladi**

- **Eng yomoni, n o'lchamdagi kirish ma'lumotlarini qayta ishlash bir xil o'lchamdagi boshqa ma'lumotlarga nisbatan eng ko'p vaqt talab qiladigan holat.**

Asimptotik tahlil: Ω()

- **Ta’rif**

Har qanday $n > n_0$ uchun $f(n) \geq cg(n)$ bo‘ladigan $c>0$ va $n_0>0$ konstanta qiymatlar mavjud bo‘lsa, musbat $f(n)$ funksiya $\Omega(g(n))$ deyiladi

- **Ma'nosi**

Kirish qiymatining eng katta qiymatlarida (n>n0), algoritm har doim cg(n) dan ortiq bosqichlarda ishlaydi.

Pastki chiziq

f(n) funksiya o‘sishi uchun Ω(g(n)) bahosi pastki asimptotik bahoni ifodalaydi va doimiy koeffitsientgacha g (n) dan sekin o'smaydigan funktsiyalar sinfini belgilaydi.

Asimptotik tahlil : Ω()

- Misol. T(*n*) = $c_1n^2 + c_2n$.

$c_1n^2 + c_2n \geq c_1n^2$ **har qanday** $n > 1$ **bo'lgan hol uchun**

T(*n*) $\geq cn^2$ $c = c_1$ **и** $n_0 = 1$

Shunday qilib, T(n) ta'rifi bo'yicha Ω(n2) dir

Barcha pastki chegaralardan eng kattasini hisoblashga qaratiladi.

Asimptotik tahlil : Θ()

- **Определение**
  - **Говорят, что неотрицательная функция $f(n)$ есть $\Theta(g(n))$, если существуют константы $c_1>0$, $c_2>0$ и $n_0>0$, такие что $c_1 g(n) \le f(n) \le c_2 g(n)$ для любых $n > n_0$.**
  - **Функция $f(n)$ есть $\Theta(g(n))$, если она одновременно есть $\Omega(g(n))$ и $O(g(n))$**

- **Смысл**
  - **Асимптотическое равенство (с точностью до константы)**
  - **Полностью описывает характер роста функции**

Асимптотический анализ

- **Правила упрощения**

1. Транзитивность
   Если $f(n)$ есть $O(g(n))$ и $g(n)$ есть $O(h(n))$, то $f(n)$ есть $O(h(n))$
2. Игнорирование констант
   Если $f(n)$ есть $O(kg(n))$ для любой константы $k > 0$, то $f(n)$ есть $O(g(n))$
3. Отбрасывание членов низких порядков
   Если $f_1(n)$ есть $O(g_1(n))$ и $f_2(n)$ есть $O(g_2(n))$, то $(f_1 + f_2)(n)$ есть $O(\max(g_1(n), g_2(n)))$
4. Мултипликативность
   Если $f_1(n)$ есть $O(g_1(n))$ и $f_2(n)$ есть $O(g_2(n))$, то $f_1(n)f_2(n)$ есть $O(g_1(n)g_2(n))$

Asimptotik tahlil

- **1-Misol**

```
a = b;
```

O’zlashtirish qiymati doimiy vaqtni oladi, shuning uchun u Θ(1) murakkablikka ega

- **2-Misol**

```
sum = 0;
for (i=1; i<=n; i++)
   sum += n;
```

Sikl chiziqli murakkablikka ega bo’lganligi uchun Θ(*n*) ga teng.

3-Misol

```
sum = 0;
for (j=1; j<=n; j++)
  for (i=1; i<=j; i++)
    sum++;
for (k=0; k<n; k++)
  A[k] = k;
```

Birinchi qator $\Theta(1)$

Ichma – ich sikllar $\Sigma i = \Theta(n^2)$

Oxirgi sikl $\Theta(n)$

Natija: $\Theta(n^2)$

- **4-Misol**

```
sum1 = 0;
for (i=1; i<=n; i++)
  for (j=1; j<=n; j++)
    sum1++;

sum2 = 0;
for (i=1; i<=n; i++)
  for (j=1; j<=i; j++)
    sum2++;
```

Birinchi juflikdagi ichma-ich sikllar– n^2 qadam

Ikkinchi juflikdagi ichma-ich sikllar – $(n+1)(n)/2$ qadam

Har ikkala juftlikda $\Theta(n^2)$

Natija: $\Theta(n^2)$

5-Misol

```
sum1 = 0;
for (k=1; k<=n; k*=2)
  for (j=1; j<=n; j++)
    sum1++;

sum2 = 0;
for (k=1; k<=n; k*=2)
  for (j=1; j<=k; j++)
    sum2++;
```

Birinchi juftlik siklda: Σn $k=1 \ldots \log n$ murakkablik $\Theta(n \log n)$

Ikkinchi juftlik siklda : $\Sigma 2^k$ $k=0 \ldots \log n - 1$ murakkablik $\Theta(n)$

Natija: $\Theta(n \log n)$