Algoritmusok bonyolultsága

2. előadás

http://www.ms.sapientia.ro/~kasa/komplex.htm

Bonyolultság mérése – bináris keresés

bináris keresés

$A_1 \leq A_2 \leq \ldots \leq A_n, \quad x$

```
Bináris_keresés(A, x)
i := 1, j := hossz[A]
while i ≤ j
        do k := ⌊(i + j)/2⌋
            if x = A_k
               then return k
            if x < A_k
               then j := k − 1
               else  i := k + 1
return 0
```

$$\begin{cases} W(n) = 1 + W\left(\left\lfloor \frac{n}{2} \right\rfloor\right), \text{ha } n > 1, \\ W(1) = 1. \end{cases}$$

megoldása: $W(n) = \lfloor \log n \rfloor + 1$, ha $n \geq 1$.

Itt a log kettes alapú logaritmust jelent.

Tétel

$W(n) = \lfloor \log n \rfloor + 1$, *ha* $n \geq 1$.

$$\begin{cases} W(n) = 1 + W\left(\left\lfloor \frac{n}{2} \right\rfloor\right), \text{ha } n > 1, \\ W(1) = 1. \end{cases}$$

Megsejtése:

$$W(n) = 1 + W\left(\left\lfloor \frac{n}{2} \right\rfloor\right) = 2 + W\left(\left\lfloor \frac{n}{2^2} \right\rfloor\right) = 3 + W\left(\left\lfloor \frac{n}{2^3} \right\rfloor\right) = \dots$$

$W(n) = \lfloor \log n \rfloor + 1$, ha $n \geq 1$. bizonyítása

Bizonyítása indukcióval:

$n = 1$-re igaz.

Indukciós feltevés: $W(k) = \lfloor \log k \rfloor + 1$ ha $1 \leq k < n$.

$$W(n) = 1 + W\left(\left\lfloor \frac{n}{2} \right\rfloor\right) = 1 + \left\lfloor \log \left\lfloor \frac{n}{2} \right\rfloor \right\rfloor + 1 = 2 + \left\lfloor \log \left\lfloor \frac{n}{2} \right\rfloor \right\rfloor =$$

$$= \begin{cases} 2 + \lfloor \log n - 1 \rfloor & \text{ha } n \text{ páros}: \quad \left\lfloor \frac{n}{2} \right\rfloor = \frac{n}{2} \\ 2 + \lfloor \log(n-1) - 1 \rfloor & \text{ha } n \text{ páratlan}: \quad \left\lfloor \frac{n}{2} \right\rfloor = \frac{n-1}{2} \end{cases}$$

$$= \begin{cases} 1 + \lfloor \log n \rfloor & \text{ha } n \text{ páros} \\ 1 + \lfloor \log(n-1) \rfloor & \text{ha } n \text{ páratlan} \end{cases}$$

De ha n páratlan, akkor $\lfloor \log n \rfloor = \lfloor \log(n-1) \rfloor$. Ezért $W(n) = \lfloor \log n \rfloor + 1$.

Bináris keresés – átlageset

Tétel

A bináris keresés algoritmusa átlagban megközelítőleg

$$\lfloor \log(n) \rfloor + \frac{1}{2}$$

összehasonlítást végez n elemű sorozat esetében.

I_i azon bemenetek halmaza, amelyekre $x = A_i$, ha $1 \leq i \leq n$,
I_{n+1} azon bemenetek halmaza, amelyekre $x < A_1$,
I_{n+i} azon bemenetek halmaza, amelyekre $A_{i-1} < x < A_i$, ha $2 \leq i \leq n$,
I_{2n+1} azon bemenetek halmaza, amelyekre $x > A_n$.
I_i esetében az összehasonlítások száma $t(I_i)$

$$A(n) = \sum_{i=1}^{2n+1} p(I_i)t(I_i)$$

Feltevések:

- $p(I_i) = \frac{1}{2n+1}$, ha $1 \leq i \leq 2n+1$,
- $n = 2^k - 1$, ahol $k \geq 1$ (azaz $k = \lfloor logn \rfloor + 1$).

Legyen s_t azon bemenetek száma, amelyek esetében az algoritmus t összehasonlítást végez ($1 \leq t \leq k$).

Ekkor

$$A(n) = \sum_{i=1}^{2n+1} p(I_i)t(I_i) = \frac{1}{2n+1} \sum_{i=1}^{k} ts_t$$

De
$s_t = 2^{t-1}$, ha $1 \leq t < k$
$s_k = 2^{k-1} + n + 1$

$n = 15$

i	$t(l_i)$	i	$t(l_i)$
1	4	17	4
2	3	18	4
3	4	19	4
4	2	20	4
5	4	21	4
6	3	22	4
7	4	23	4
8	1	24	4
9	4	25	4
10	3	26	4
11	4	27	4
12	2	28	4
13	4	29	4
14	3	30	4
15	4	31	4
16	4		

$s_1 = 1, \quad s_2 = 2, \quad s_3 = 4, \quad s_4 = 8 + 16 = 24$

$$A(n) = \frac{1}{2n+1}\sum_{i=1}^{k} ts_t = \frac{1}{2n+1}\left(\sum_{t=1}^{k} t2^{t-1} + k(n+1)\right)$$

Számítsuk ki:

$$\sum_{t=1}^{k} t2^{t-1} = \sum_{t=1}^{k} t(2^t - 2^{t-1}) = \sum_{t=1}^{k} t2^t - \sum_{t=0}^{k-1}(t+1)2^t =$$

$$= (k-1)2^k + 1$$

$$A(n) = \frac{(k-1)2^k+1}{2n+1} + \frac{k(n+1)}{2n+1} = \frac{(k-1)2^k+1}{2\cdot 2^k - 1} + \frac{k2^k}{2\cdot 2^k - 1}$$

$$A(n) \approx \frac{k-1}{2}\cdot\frac{k}{2} = k - \frac{1}{2} = \lfloor \log n \rfloor + \frac{1}{2}$$

integrálással

$$f(x) = \sum_{t=1}^{k} tx^{t-1}$$

$$F(x) = \int f(x)dx = \frac{x^{k+1} - x}{x - 1} + C$$

$$f(x) = F'(x) = \frac{kx^{k+1} - (k+1)^k + 1}{(x-1)^2}$$

$$f(2) = \sum_{t=1}^{k} t2^{t-1} = (k-1)2^k + 1$$

i	$t(I_i)$	i	$t(I_i)$
1	4	17	4
2	3	18	4
3	4	19	4
4	2	20	4
5	4	21	4
6	3	22	4
7	4	23	4
8	1	24	4
9	4	25	4
10	3	26	4
11	4	27	4
12	2	28	4
13	4	29	4
14	3	30	4
15	4	31	4
16	4		

$n = 15$: átlag=3.65, elméleti átlag= 3.50

i	$t(I_i)$	i	$t(I_i)$	i	$t(I_i)$	i	$t(I_i)$
1	5	17	5	33	5	49	5
2	4	18	4	34	5	50	5
3	5	19	5	35	5	51	5
4	3	20	3	36	5	52	5
5	5	21	5	37	5	53	5
6	4	22	4	38	5	54	5
7	5	23	5	39	5	55	5
8	2	24	2	40	5	56	5
9	5	25	5	41	5	57	5
10	4	26	4	42	5	58	5
11	5	27	5	43	5	59	5
12	3	28	3	44	5	60	5
13	5	29	5	45	5	61	5
14	4	30	4	46	5	62	5
15	5	31	5	47	5	63	5
16	1	32	5	48	5		

$n = 31$: átlag = 4.59, elméleti átlag = 4.50

i	$t(I_i)$	i	$t(I_i)$	i	$t(I_i)$	i	$t(I_i)$
1	4	11	4	21	4	31	4
2	3	12	3	22	4	32	4
3	4	13	4	23	4	33	4
4	5	14	5	24	5	34	5
5	2	15	2	25	5	35	5
6	4	16	4	26	4	36	4
7	3	17	5	27	4	37	5
8	4	18	3	28	4	38	5
9	5	19	4	29	5	39	4
10	1	20	5	30	5	40	5
						41	5

n=20: átlag = 4.10, elméleti átlag = 4.50

Optimalitás

- A algoritmus, legrosszabb esetben $W(n)$ művelet
- minden algoritmus egy adott osztályból legalább $F(n)$ műveletet végez

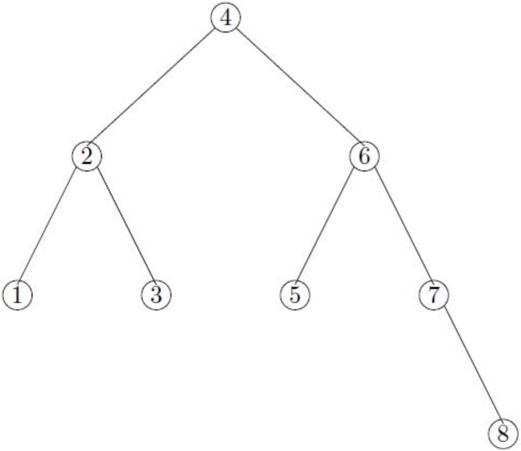
Ha $F(n) = W(n)$, minden n-re, akkor az A algoritmus optimális az adott osztályon belül.

Max(A)

```
max := A_1
for i := 2 to hossz[A]
    do if max < A_i
          then max := A_i
return max
```

$W(n) = n - 1$

Legnagyobb elem keresésére legalább $n - 1$ művelet szükséges. Tehát a Max algoritmus optimális.



d magasságú bináris fában a csúcsok száma legfeljebb $2^{d+1} - 1$

döntési fa, bináris keresés, $n = 8$

Tetszőleges d magasságú keresési algoritmus legrosszabb esetben $d+1$ összehasonlítást végez. Ha n elemünk van, akkor mind az n elemnek kell szerepelnie legalább egyszer a döntési fában. Tehát

$$n \leq 2^{d+1} - 1, \text{ ahonnan } d+1 \geq \log(n+1),$$

$$d \geq \lceil \log(n+1) \rceil - 1,$$

ekkor az összehasonlítások száma legalább

$$\lceil \log(n+1) \rceil = \lfloor \log n \rfloor + 1.$$

Mivel a keresési algoritmus legrosszabb esetben $\lfloor \log n \rfloor + 1$ öszehasolítást igényel, ezért ez az algoritmus optimális.

Buborékrendezés bonyolultsága

Buborékrendezés(A)

```
n := hossz[A]
k := n; ind := 1
while ind > 0
        do k := k − 1
           ind := 0
           for i := 1 to k
               do if A_i > A_{i+1}
                      then A_i ⇔ A_{i+1}
                           ind := 1
return A
```

$$
\begin{aligned}
W(n) &= \\
&= (n-1) + (n-2) + \ldots + 1 \\
&= \frac{(n-1)n}{2} \\
&= \Theta(n^2)
\end{aligned}
$$

$$A(n) \approx \frac{n^2}{4} = \Theta(n^2)$$

Knuth, 3. kötet, 5.1.1. szakasz

Gyorsrendezés bonyolultsága

Gyorsrendezés(A, b, j)

1. **if** $b < j$
2. **then** $k :=$ Feloszt(A, b, j)
3. Gyorsrendezés$(A, b, k-1)$
4. Gyorsrendezés$(A, k+1, j)$
5. **else** **return** A

Felosztáskor egy elem (a k-adik) a helyére kerül. Legrosszabb esetben a felosztás úgy működik, hogy eredményként egyik halmaz üres lesz. Ekkor az összehasonlítások száma:

$$(n-1) + (n-2) + \ldots + 1 = \frac{n(n-1)}{2} \Longrightarrow W(n) = \Theta(n^2)$$

Feltevés: a felosztás eredménye két egyforma nagyságú rész
$n = 2^p - 1$ eset ($Q(p)$ összehasonlítások száma):

$$Q(p) = 2^p - 2 + 2Q(p-1)$$

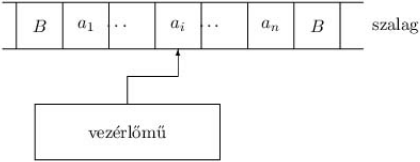
$$Q(1) = 0$$

$$Q(p) = \sum_{i=1}^{p-1} (2^p - 2^i) = (p-1)2^p - \sum_{i=1}^{p-1} 2^i$$

$$= (p-1)2^p - (2^p - 2) = \lfloor \log n \rfloor (n+1) - n + 1$$

Ebben az esetben $A(n) = \Theta(n \log n)$.
Általában ez kb. $2n \log n$.

Turing-gép



Determinisztikus Turing-automatának vagy *determinisztikus Turing-gépnek* nevezzük a következő rendezett hetest:

$$T = (Q, \Sigma, W, \delta, q_0, B, F),$$

ahol

- Q a belső állapotok véges és nem üres halmaza,
- Σ a bemeneti ábécé,
- W a szalagábécé ($\Sigma \subseteq W$),
- $\delta : Q \times W \to Q \times W \times \{-1, +1\}$, átmenetfüggvénynek nevezett leképezés,
- $q_0 \in Q$ a kezdőállapot,
- $B \in W \setminus \Sigma$ a blank jele (szóköz, üres hely),
- $F \subseteq Q$ a végállapotok halmaza.

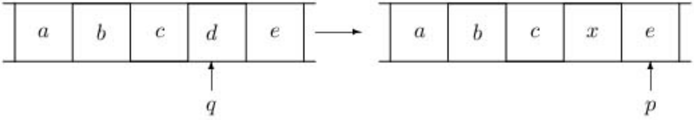
Az átmenetfüggvényben a -1 a balra való elmozdulást, míg a $+1$ a jobbra való elmozdulást jelenti. Szokás még -1 helyett az L betűt (left), $+1$ helyett pedig az R betűt (right) használni. Amennyiben az átmenetfüggvény

$$\delta \,:\, Q \times W \to \mathcal{P}(Q \times W \times \{-1, +1\}),$$

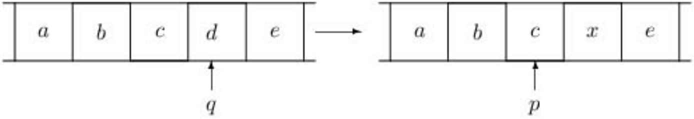
az automata *nemdeterminisztikus*. Ha csak Turing-automatát írunk, mindig determinisztikust értünk rajta.

Turing-gép

Példa átmenetekre:



$$(abc, q, de) \Longrightarrow (abcx, p, e), \quad \text{ha} \quad \delta(q, d) = (p, x, +1)$$



$$(abc, q, de) \Longrightarrow (ab, p, cxe), \quad \text{ha} \quad \delta(q, d) = (p, x, -1)$$

A T Turing-automata felismeri az u szót, ha létezik egy $(\varepsilon, q_0, u) \overset{*}{\underset{T}{\Longrightarrow}} (\alpha, q, \beta)$ teljes számítási folyamat, ahol $q \in F$. Az üres szót akkor ismeri fel, ha a kezdő konfiguráció (ε, q_0, B) és $q_0 \in F$. A T Turing-automata által felismert szavak halmazát $L(T)$-vel jelöljük.

Turing-gép – példa

Példa:

$$T_1 = (\{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}, \{0, 1\}, \{0, 1, B\}, \delta, q_0, B, \{q_0\}).$$

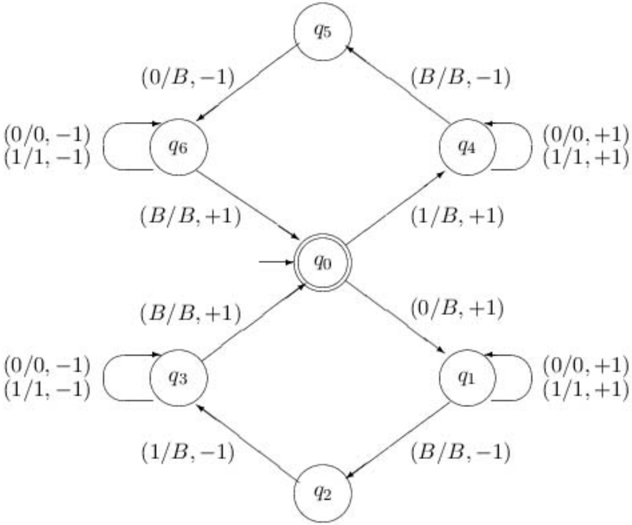
Az átmenetfüggvény:

$$
\begin{array}{lll}
\delta(q_0, 0) = (q_1, B, +1), & \delta(q_0, 1) = (q_4, B, +1), & \\
\delta(q_1, 0) = (q_1, 0, +1), & \delta(q_1, 1) = (q_1, 1, +1), & \delta(q_1, B) = (q_2, B, -1), \\
\delta(q_2, 1) = (q_3, B, -1), & & \\
\delta(q_3, 0) = (q_3, 0, -1), & \delta(q_3, 1) = (q_3, 1, -1), & \delta(q_3, B) = (q_0, B, +1), \\
\delta(q_4, 0) = (q_4, 0, +1), & \delta(q_4, 1) = (q_4, 1, +1), & \delta(q_4, B) = (q_5, B, -1), \\
\delta(q_5, 0) = (q_b, B, -1), & & \\
\delta(q_6, 0) = (q_6, 0, -1), & \delta(q_6, 1) = (q_6, 1, -1), & \delta(q_6, B) = (q_0, B, +1).
\end{array}
$$

Átmenettáblázat:

	0	1	B
q_0	$(q_1, B, +1)$	$(q_4, B, +1)$	–
q_1	$(q_1, 0, +1)$	$(q_1, 1, +1)$	$(q_{@}, B, -1)$
q_2	–	$(q_3, B, -1)$	–
q_3	$(q_3, 0, -1)$	$(q_3, 1, -1)$	$(q_0, B, +1)$
q_4	$(q_4, 0, +1)$	$(q_4, 1, +1)$	$(q_5, B, -1)$
q_5	$(q_6, B, -1)$	–	–
q_6	$(q_6, 0, -1)$	$(q_6, 1, -1)$	$(q_0, B, +1)$

Átmenetdiagram:



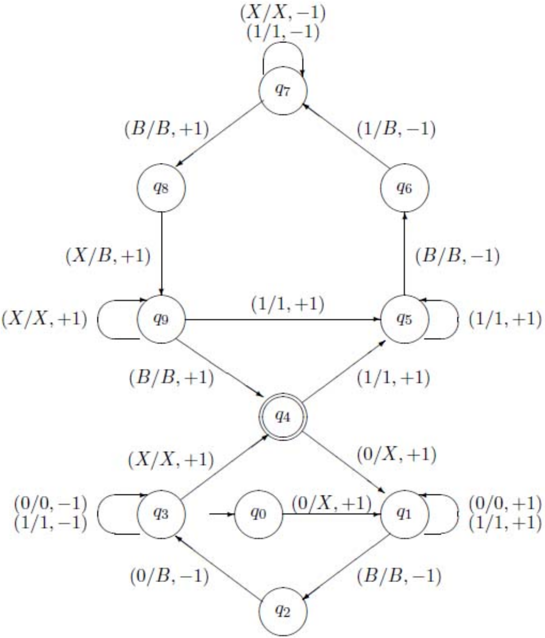
Turing-gép – példa

Bemutatunk néhány konkrét példát az automata működésére. Megmutatjuk, hogy $001011 \in L(T_1)$, $001 \notin L(T_1)$, $0100 \notin L(T_1)$.

$(\varepsilon, q_0, 001011) \Longrightarrow (\varepsilon, q_1, 01011) \Longrightarrow (0, q_1, 1011) \Longrightarrow (01, q_1, 011) \Longrightarrow (010, q_1, 11) \Longrightarrow$
$(0101, q_1, 1) \Longrightarrow (01011, q_1, B) \Longrightarrow (0101, q_2, 1) \Longrightarrow (010, q_3, 1) \Longrightarrow (01, q_3, 01) \Longrightarrow$
$(0, q_3, 101) \Longrightarrow (\varepsilon, q_3, 0101) \Longrightarrow (\varepsilon, q_3, B0101) \Longrightarrow (\varepsilon, q_0, 0101) \Longrightarrow (\varepsilon, q_1, 101) \Longrightarrow$
$(1, q_1, 01) \Longrightarrow (10, q_1, 1) \Longrightarrow (101, q_1, B) \Longrightarrow (10, q_2, 1) \Longrightarrow (1, q_3, 0) \Longrightarrow$
$(\varepsilon, q_3, 10) \Longrightarrow (\varepsilon, q_3, B10) \Longrightarrow (\varepsilon, q_0, 10) \Longrightarrow (\varepsilon, q_4, 0) \Longrightarrow (0, q_4, B) \Longrightarrow$
$(\varepsilon, q_5, 0) \Longrightarrow (\varepsilon, q_6, B) \Longrightarrow (\varepsilon, q_0, B)$
Nincs tovább átmenet, q_0 végállapot, ezért az automata felismeri a 001011 szót.

$(\varepsilon, q_0, 0100) \Longrightarrow (\varepsilon, q_1, 100) \Longrightarrow (1, q_1, 00) \Longrightarrow (10, q_1, 0) \Longrightarrow (100, q_1, B) \Longrightarrow$
$(10, q_2, 0)$.
Nincs átmenet, q_2 nem végállapot, tehát T_1 nem ismeri fel a 0100 szót.

A T_1 Turing-automata az $a_1 a_2 \ldots a_n b_1 b_2 \ldots b_n$ alakú szavakat ismeri fel, ahol $a_i \in \{0, 1\}$, $b_i = 1 - a_{n+1-i}$, $i = 1, 2, \ldots, n$.



$$L(T_2) = \left\{0^n 1^n 0^n \mid n > 0\right\}$$

Turing-automaták változatai

A következőkben bemutatjuk a Turing-automaták néhány változatát. A felismerés szempontjából ezek mind ekvivalensek az eredetileg értelmezett Turing-automatával.

1) *Többszalagos Turing-automata*

Az automatának k darab szalagja van, mindegyik saját író/olvasó fejjel. Az átmenetfüggvény:

$$\delta : Q \times W^k \to Q \times W^k \times \{-1, +1\}^k$$

Egy többszalagos Turing-automatához mindig hozzárendelhető egy vele ekvivalens egyszalagos Turing-automata.

2) *Nemdeterminisztikus Turing-automata*

Mint már láttuk, a nemdeterminisztikus Turing-automata átmenetfüggvénye a következő:

$$\delta : Q \times W \to \mathcal{P}(Q \times W \times \{-1, +1\}).$$

Tetszőleges nemdeterminisztikus Turing-automatához megadható egy vele ekvivalens determinisztikus Turing-automata.

3) *Helybenlépéses Turing-automata*

Megengedjük, hogy átmenetkor az író/olvasó fej helyben is maradhasson. Azaz az átmenetfüggvény a következő:

$$\delta : Q \times W \to Q \times W \times \{-1, 0, +1\}$$