ALGORITMUSELMÉLET
ÖSSZEFOGLALÓ

$log\ n \leq n^2 \leq 2^n \leq n! \leq n^n$

1. **Keresés**
   - ***Lineáris*** (lépésszám: $O(n)$)
   - ***Bináris*** (lépésszám: $O(log\ n)$) – optimális

2. **Rendezés**
   - ***Buborék*** (lépésszám: $O(n^2)$ – $max\ n * \frac{n-1}{2}$)
   - ***Beszúrásos*** (lépésszám: lineáris keresésnél: $O(n^2)$; bináris keresésnél: $O(n\ log\ n)$)
   - ***Összefésülés*** (lépésszám: k+l-1) – összefésüléses rendezés (lépésszám: $O(n\ log\ n)$)
   - ***Quick Sort*** (lépésszám: $\Theta(n^2)$)
     Átlagosan: $O(n\ log\ n$
   - ***Láda*** (lépésszám: $O(n + |A|)$)
   - ***Radix*** (lexikografikus) (lépésszám: $O(kn + \Sigma|A_i|)$)
     a. $|A_i| \leq cn$ – $O(kn)$
     b. $|A_i| \leq c$ – $O(n\ log\ n)$

3. **Adatszerkezetek**
   - ***Bináris fa***
     Teljes bináris fa
   - ***Kupac***
     Kupac tulajdonság: 1 csúcsbeli elem< balfia és a jobbfia
     Bináris kupac $(d = 2)$
       Feltétel: a tárolt elemek különbözőek
       FELSZIVÁROG(F): $O(log\ n)$ – F: teljes bifa
       MINTÖR: $O(log\ n)$
         Legkisebb elem a gyökérben van, ezt töröljük, a legutolsó értéket berakjuk helyére. (és aztán Felszivárog())
       BESZÚR: $O(log\ n)$
       Kupacépítés: $O(n)$
     d-kupacok
       szintek száma: $\Theta(\log_d n)$
       FELSZIVÁROG: $O(d \log_d n)$
       MINTÖR: $O(d \log_d n)$
       BESZÚR: $O(\log_d n)$
   - ***Keresőfák***
     Keresőfa tulajdonság: minden x csúcsra teljesül, hogy
     baloldali részfájában lévő elemek < x-beli elem < jobb oldali részfa eleme
     - ***Bináris keresőfa***
       $n \geq$ szintszám n elem esetén $\geq \log n$
       Bejárás: preorder (gy, F_1, F_2), inorder (F_1, gy, F_2), postorder (F_1, F_2, gy)
       Inordernél növekvő sorrend!
       Faépítés átlagos lépésszáma: $O(n\ log\ n)$; átlagos szintszáma: $\Theta(log\ n)$
       KERES(x): $O(szintszám)$
         x (?) gyökér = kész; < -bal; > -jobb részfában keresünk
       MIN: $O(szintszám)$
         Balra lép, amíg tud.
       MAX: $O(szintszám)$
         Jobbra lép, amíg tud.
       TÓLIG(a,b): $O(szintszám +\ a\ és\ b\ között\ elemek\ száma)$
       BESZÚR(x): $O(szintszám)$
         Ha már van x, nem kell beszúrni, ha nincs, ott hozzuk létre, hogy a keresés elakadt.
       TÖRÖL(x): $O(szintszám)$
         Ha nincs ilyen elem, kész.
         Ha megtaláltuk és
           x levél: simán töröljük
           1 fia van: kihagyjuk x-et, a fiát rakjuk a helyére
           2 fia van: bal r.fájának max elemét rakjuk helyére
     - ***Piros-fekete fa***
       Gyökér, levelek: feketék

Piros csúcs fiai feketék

Minden v csúcsra: az összes v-ből a levelekhez vezető úton uannyi fekete csúcs van

v csúcs magassága ($m(v)$): v-ből levélbe vezető úton élek max. száma

fekete magassága ($fm(v)$): v-ből levélbe vezető úton a fekete csúcsok száma

$$\frac{m(v)}{2} \leq fm(v) \leq m(v)$$

„n” elem esetén: fa magassága: $\leq 2\, log\, (n+1)$

KERES(x): $O(log\, n)$

MIN: $O(log\, n)$

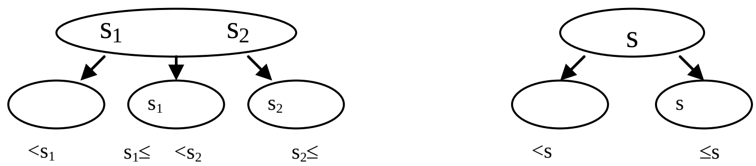
MAX: $O(log\, n)$

BESZÚR(x): $O(log\, n)$, közben ≤ 2 forgatás történik

TÖRÖL(x): $O(log\, n)$, összesen ≤ 3 forgatás történik

- ***2-3 fa (m=3)***

A levelek azonos szinten vannak, levelekben az elemek balról jobbra növekvő sorrendben. (elemek csak a levelekben)



$\log_3 n \leq magassága \leq log\, n$

levelek száma: $2^m - 1$ (m: szintszám)

KERES(x): $\Theta(log\, n)$

MIN: $O(log\, n)$ – első levél

MAX: $O(log\, n)$ – utolsó levél

TÓLIG(a,b): $O(log\, n\ +\ a - b\ intervallum\ hossza)$

BESZÚR(x): $O(log\, n)\ =\ O(magasság)$

Keres() meghat. a helyét; ha az apjának eddig 2 fia volt – 3 lesz

3 fia volt – csúcsvágás

Csúcsvágás: 1 szinttel feljebb az új csúcsot beilleszteni ugyanígy. Ha ez felmegy a gyökérig, ott is kell vágni, új gyökér, a fa magassága nő.

TÖRÖL(x): $O(log\, n)$

Ha nem találtuk meg → nincs törlés

Ha a törlendő csúcs apjának

3 gyereke van – töröljük a csúcsot

2 gyereke van

van 3 gyerekes testvér: megosztoznak a maradék 4 gyereken

nincs: csúcsösszevonás – 1szinttel feljebb törlés

- ***B-fa***

B_m fa magassága („n” db levél): $\Theta(\log_m n)$

$\left[\frac{m}{2} + 0{,}5\right] \leq belső\ csúcs\ fiainak\ száma \leq m$

$2 \leq gyökér\ fiainak\ száma \leq m$

KERES(x)

BESZÚR(x) – csúcsvágás

TÖRÖL(x) – csúcsösszevonás

– ***Hash***

- ***Vödrös hash-elés***

$V[0..H-1]$ – vödörkatalógus (vödör: rendezett/rendezetlen láncolt lista)

$h: S \to \{0,1,\dots,n-1\}$

KERES(t): $V[h(t)]$ listában lin. kereséssel; $O(n)$

BESZÚR(t): lista elejére (rendezetlen); listába, a helyére (rendezett)

TÖRÖL(t): kb. mint a Beszúr(); $O(n)$

- ***Nyitott címzésű hash***

$T[0..M-1]$ – hash tábla

Lineáris próbálás – $O(M)$

Kvadratikus próbálás

- ***Osztó/szorzó módszer***

- ***Kettős hashnél:***

$h'(s) = (s\, mod\, M - 1) + 1$

⊕ nincs benne rendezés

⊖ MIN, MAX, TÓLIG keresésre nem jó
⊖ betelhet a hash-tábla

4. Gráf algoritmusok

- ***Megadás: éllista, súlyozott gráf, szomszédossági mátrix***
- ***Bejárás***
  Szélességi – BFS (éllistával adott: lépésszám: $O(n+e)$)
  Mélységi – DFS (lépésszám: éllistás: $O(n+e)$; mátrixos: $O(n^2)$)
- ***Legrövidebb út keresése***
  Szélességi bejárás
  Dijkstra:$O(n^2)$ – v→minden más (minden él>=0)
  KÉSZ halmazba gyűjtjük azokat a pontokat, amelyekről már tudjuk, hogy v-től milyen távolságra vannak
  Bellman-Ford: $O(n^3)$ – v→minden más (nincs negatív kör)
  Max 1,2,3… hosszú úton legrövidebb út keresés
  $T[k,i] = Min([T(k-1),1],[T(k-1),2]+E(2,i),\dots[T(k-1),n]+E(n,i))$
  ∟ $v_1 \rightarrow v_i$-be legfeljebb k hosszú legrövidebb út
  Floyd: $O(n^3)$ – v→minden pontba (nincs negatív kör)
  Max 0,1,2… ponton keresztül legrövidebb út keresés
  $T^k+1(i,j) = Min\{T^k[i,k+1]+T^k[k+1,j],T^k(i,j)\}$
- ***Egyéb***
  DAG: irányított gráf, nincs benne irányított kör -> legrövidebb, leghosszabb út keresés
  G DAG ⇔ egy mélységi bejárásban nincs visszaél
  G-nek van topológikus rendezése ⇔ G DAG
  Topologikus rendezés: csak a kisebb sorszámú pontokból a nagyobbak felé megy él
  Tranzitív lezárt: Az eredeti élek benne vannak a gráfban + behúzunk még olyan élet, ami x-ből y-ba mutat, ha van olyan út, amin el lehet jutni x-ből y-ba.
  Faél: olyan éle a gráfnak, ami a bejáráskor új csúcsba vitt (faélek fesz.erdőt alkotnak)
  Párosítás: egy G irányítatlan gráfban=fgtlen élek száma
  Páros gráfban max. párosítás keresése: magyar módszer (lépésszám: $O(n\ (n+e))$)
- ***Piros-kék algoritmus: minimális súlyú feszítőfa keresése (kék élek)***
  Kék szabály: x-ből nincs kivezető kék él, akkor egy legkisebb x-ből kivezető színtelen élet kékre színezni
  Piros szabály: C egy kör a gráfban, ha még nincs piros éle, egy legnagyobb súlyú színtelen élét pirosra színezzük
  Takaros színezés: ha van olyan fesz.fa, ami az összes kék élet tartalmazza és egyetlen pirosat sem

  Prim-Jarnik (lépésszám: $O(n^2)$; éllistával: $O(e\ log\ n)$)
  Vegyünk egy pontot és a kékszabállyal csináljunk belőle fesz.fát.
  Kruskal (lépésszám: $O(e\ log\ e)$)
  Vegyük az éleket növekvő sorrendbe és nézzük meg, hogy a következő él behúzásával (kék) kört kapunk-e, ha nem, húzzuk be (kék), ha igen, akkor ne (piros)
  UNIÓ-HOLVAN tömbbel (lépésszám: $O(e\ log\ n+n^2)$)
  UNIÓ-HOLVAN fával (lépésszám: $O(e\ log\ n)$)
  $O(e)$ lépésben eldönthető, hogy egy adott feszítőfa ***minimális-e***

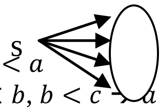
5. Bonyolultság elmélet

Polinom időben megoldható(***P***)/nem megoldható(***NP***).
Karp redukció (visszavezetés): a mi problémánkat visszavezetjük egy ismert NP-teljes problémára; a mi problémánk nehezebb, mint az NP-teljes probl.
P € NP-teljes, ha
NP-ben van: igen válaszra van polinom méretű, P időben ellenőrizhető tanu
NP-nehéz: minden NP-beli probléma visszavezethető rá
Ismert NP-teljes problémák:
3SZÍN: 3 színnel színezhető gráf (4SZÍN stb.)
MAXFTL(G,k): G gráfban van k db független pont
MAXKLIKK(G,k): G-ben van k pontú teljes részgráf
HAMILTON-KÖR: G-ben van H-kör
HAMILTON-ÚT: G-ben van H-út
GRÁFIZO(G_1,G_2): G_1 izomorf G_2-vel (élet élbe,nemélet nemélbe visz)
2DH: páros gráfban van-e teljes párosítás
3DH: 3 dimenziós házasítás
X2C: teljes párosítás általános gráfban
RHÖSSZEG(H,x): H-nak van olyan részhalmaza, amelynek összege x
HÁTIZSÁK($\{a_1..a_k\}$értékek, $\{s_1..s_k\}$súlyok, S-teherbírás, E-célérték)

Definíciók:

Szuperforrás: bele nem megy él, belőle mindenhova megy él. Csak 1 szuperforrás lehet egy gráfban!!!

< reláció:



Irreflexív: a ! $< a$

Tranzitív: $a < b$, $b < c \rightarrow a < c$

Teljes: $a < b$ vagy $b < a$ vagy $a = b$

BSZ2 emlékeztető☺:

síkbarajzolható a gráf: ha nem tartalmaz $K_{3,3}$-at és K_5-öt topologikus részgráfként.

G gráfban ***van Euler-kör*** ⇔ minden fokszám páros és összefüggő a gráf