Midterm
- matching problems
- order of growth fxns
- algorithims on graphs

Jul 16 180

Merge-Sort $ms(A, B)$ $A = (a_1, a_2, \dots, a_n)$ ordered
$B = (b_1, b_2, \dots, b_n)$

c is the current pointer

Put $A_0 = A$, $B_0 = B$, $c(A_0) = a_1$, $c(B_0) = b_1$, $i = 0$, $D_0 = \emptyset$
While $A_i \neq \emptyset$ & $B_i \neq \emptyset$
  choose $\min(c(A_i), c(B_i)) = c$
  If $c = a_t$, then $A_{i+1} = A_i \setminus \{a_t\}$
    $B_{i+1} = B_i$
    $D_{i+1} = (D_i, a_t)$
  else $(c = b_m)$
    $B_{i+1} = B_i \setminus \{b_m\}$
    $A_{i+1} = A_i$
    $D_{i+1} = (D_i, b_m)$
End While
If $A_i = \emptyset$, then $D_{i+1} = (D_i, B_i)$
else $D_{i+1} = (D_i, A_i)$

Example $A = (2, 7)$, $B = (1, 5)$

$A_0 = (2,7)$, $B_0 = (1,5)$, $c(A_0) = 2$, $c(B_0) = 1$, $i=0$, $D_0 = \emptyset$

$c = 1$ $A_1 = (2,7)$ $B_1 = 5$ $D_1 = (1)$

$c(A_1) = 2$ $c(B_1) = 5$ $c = 2$

$A_2 = (7)$ $B_2 = (5)$ $D_2 = (1, 2)$

$c(A_2) = 7$, $c(B_2) = 5$ $c = 5$

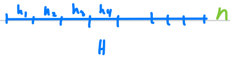
$A_3 = 7$ $B_3 = \emptyset$ $D_3 = 1, 2, 5$

$D_4 = 1, 2, 5, 7$

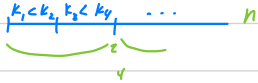
$5n = O(n)$

$H = (h_1, h_2, \dots, h_n)$

$n = 2^m$



$2n = O(n)$



$t \to t+1$

m levels

$O(A) = O(m \cdot n) = O(n \log n)$

$2^{m-1} < n \leq 2^m$

$\log_2 n$ $\quad m = \lceil \log_2 n \rceil$

Prop 4 $T_{isp}(n) = O(n \log n)$

Ordered $[f(r_1), f(r_2), \dots, f(r_n)]$

$[\underbrace{s(g_1), s(g_2)}, \dots, s(g_n)]$

$\leq 2 g$

$A_1 = \{r_1\}$ $\quad R_1 = R \setminus \{g_1, g_2\}$ $\quad O(n)$

$2\, O(n \log_2 n) + O(n) = O(n \log_2 n)$

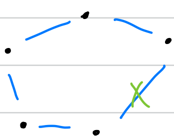
$G = (V, E)$ $\quad$ cost: $E \to \mathbb{R}^+$

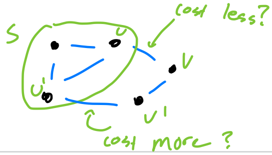
connected

minimum spanning Tree

Lemma 1 If H is a subgraph of G & H is min & H is sp, then H is a tree

Proof If H is not a tree, then $\exists$ cycle in H





Lemma 2 If $S \subseteq V$ & $e = (u, v)$ s.t. $u \in S$, $v \notin S$

then $\forall$ min sp tree T contains e

Proof Let $T = (V, D)$ sp tree $\Rightarrow \exists$ path p in T from u to v $\exists$ $e' \in p$ s.t. $u' \in S$ & $v' \notin S$

(u', v')

$T' = (V, D')$ $\quad D' = (D \setminus \{e'\}) \cup \{e\}$

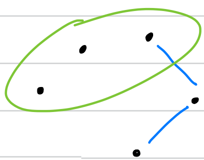
sp tree $\quad$ cost $D' <$ cost D b/c

cost of e is $<$ cost of e'

Kruskal's algorithm

Add edges in order of increasing cost if they don't create cycles

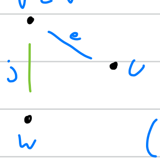
$G \xrightarrow{KA} H$



Prim's Algorithm

$v \in V$ $\quad$ Choose v & post $S_0 = \{v\}$

$S_0 = \{v, u\}$



S_{n-1}

(V, H)

Reverse-Delete algorithm

Delete the most expensive edge that does not make the graph disconnected