

$$\begin{aligned}
 T(k+1) &\leq C(k+1) \log_2 \left(\frac{k+1}{2} \right) + O(k+1) \\
 &\leq C(k+1) (\log_2(k+1) - 1) + a(k+1) \\
 &= C(k+1) \log_2(k+1) + \frac{(a-C)(k+1)}{2}
 \end{aligned}$$

if we make $C \geq a$

$$\Rightarrow T(k+1) \leq C(k+1) \log_2(k+1)$$

Conclusion: ...

Substitution:

assume $T(n) = O(n \log n) \Rightarrow T(n) \leq C n \log n$
 this is true if we can prove that C always exists for large n .

Induction (Base) $T(2) \leq C \cdot 2 \cdot \log 2 = 2C$ this is always true because $T(2)$ is a constant

(induction) $T(k) \leq Ck \log k$ for $2 \leq k$

$$T(k+1) = 2T\left(\frac{k+1}{2}\right) + O(k+1) \leq 2 \cdot C \frac{(k+1)}{2} \log \frac{(k+1)}{2} + O(k+1)$$

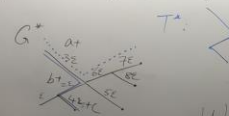
SHOT ON MI 8
AI DUAL CAMERA

relax the constraint that G only has distinct-weighted edges.



T (the hypothetical MST of G)

$$W_G(T^*) = a + b + c$$



$$W_{G^*}(T^*) = a + b + c$$

$$W_G(T) = a + d + (3e + 7e + 6e) \leq a + d + (8 + 28 + 36 + 48e) = a + d + 82e = W_G(T) + 82e$$

A : Kruskal's algorithm

$$W_{G^*}(T^*) = (a + 3e) + (b + 2e) + (c + 4e)$$

SHOT ON MI 8
AI DUAL CAMERA

$$W(T^*) - W(T) < \sum_{i=1}^{|E|} \epsilon_i$$

$$W_{G^*}(T^*) \leq W_{G^*}(T)$$

$$W_{G^*}(T) \leq W_G(T) + \epsilon \sum_{i=1}^{|E|} \epsilon_i$$

$$W_{G^*}(T^*) \leq W_G(T) + \epsilon \sum \epsilon_i$$

$$W_{G^*}(T^*) - W_G(T) \leq \epsilon \sum \epsilon_i$$

provided that

flow exists at least
one pair of
distinctly-weighted edges

SHOT ON MI 8
AI DUAL CAMERA

Kruskal's algo.

Sort edges by weights

for $i = 1 \dots |E|$

$(u, v) = e_i$ / $O(1)$

if $(u$ and v are in different sets) / $O(1)$

$T = T \cup \{e_i\}$ / $O(1)$

merge the connected components containing u and v / $O(\log |V|)$

endif

endfor

return T

Two arrays

Sae	Set
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22
23	23
24	24
25	25
26	26
27	27
28	28
29	29
30	30
31	31
32	32
33	33
34	34
35	35
36	36
37	37
38	38
39	39
40	40
41	41
42	42
43	43
44	44
45	45
46	46
47	47
48	48
49	49
50	50
51	51
52	52
53	53
54	54
55	55
56	56
57	57
58	58
59	59
60	60
61	61
62	62
63	63
64	64
65	65
66	66
67	67
68	68
69	69
70	70
71	71
72	72
73	73
74	74
75	75
76	76
77	77
78	78
79	79
80	80
81	81
82	82
83	83
84	84
85	85
86	86
87	87
88	88
89	89
90	90
91	91
92	92
93	93
94	94
95	95
96	96
97	97
98	98
99	99



Combine sets containing
 u and v .



$$\{ \cdot \} \{ \cdot \} \{ \cdot \} \Rightarrow \{ u, v \}$$

SHOT ON MI 8
AI DUAL CAMERA

Proposition: Relabeling can be done in $\log(V)$ time on average.

Proof: We choose to rename the set with a smaller size.

* for each vertex.

each merge operation must at least double the size of set that contains it.

because each set can contain at most $|V|$ elements, there are at most $\log(V)$ renaming for each vertex.

* Total # of renaming: $|V| \cdot \log(V)$

Since there are at most $|V|$ merges, each merge takes $O(\log(V))$ on average.

$$X = 100$$

$$y = 20$$

\Downarrow 20 renaming

$$X \cdot y = 120$$

SHOT ON MI 8
AI DUAL CAMERA

Divide-and-Conquer Algorithms

Merge-Sort.

given a list of numbers

Separate the list into two sets, each with half size.

$T(1)$

$T(\frac{n}{2})$

$T(\frac{n}{2})$

$O(n)$

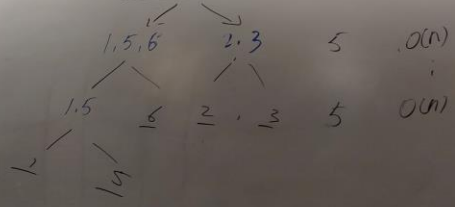
return S

MergeSort(S_1)

MergeSort(S_2)

Combine S_1 and S_2 into S.

$S = 1, 2, 3, 5, 6$

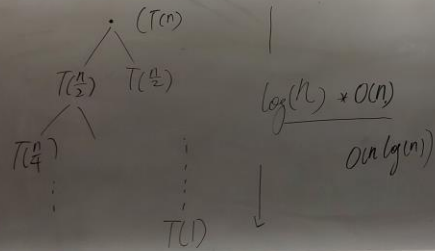


SHOT ON MI 8
AI DUAL CAMERA

Time complexity:

$$T(n) = 2(T(\frac{n}{2})) + O(n)$$

Unrolling:



Substitution:

assume $T(n) = O(n \log n) \Rightarrow T(n) \leq C n \log(n)$
 this is true if we can prove that C always exists for
 large n .

Induction (Base) $T(2) \leq C \cdot 2 \cdot \log 2 = 2C$ this is always true
 because $T(2)$ is a constant

(induction) $T(k) \leq C k \log k$ for $2 \leq k$

$$T(k+1) = 2T(\frac{k+1}{2}) + O(k+1) \leq 2 \cdot C \frac{(k+1)}{2} \log(\frac{k+1}{2}) + O(k+1)$$