Better your Bounds

```
for i = 1 to n
  for j = i + 1 to n
    if (arr[i] > arr[j]) count += 1
```

$$\sum_{i=1}^{n} (n-i) = \sum_{i=1}^{n} n - \sum_{i=1}^{n} i$$

$$= n^2 - \frac{n(n-1)}{2}$$

$$\Theta(n^2)$$

running time for a loop

$$\sum_{\text{iteration } i} \text{work(iteration } i) \le \sum (n-1)$$

$$= n(n-1)$$

$$\text{work(iteration } i) \le n-1 \quad \Theta(n^2)$$

$$\text{work}(u) \le n$$

```
for each vertex u
  for each v in Adj[u]
    add u to Adj'[v]
```

$\Theta(1)$

process (u,v)

i = iterator through Adj[u]
while (i.has next()
  v = i.next()
  add u to Adj'[v]

$$\sum_{u \in V} \text{work(iteration } u) \le \sum_{u \in V} n = n \cdot n$$

$$= n^2$$

$$O(n^2)$$

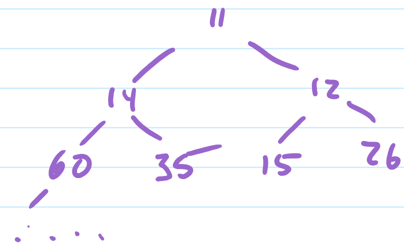
$$\sum_{u \in V} 1 + \text{degree}(u)$$

$$= \sum_{u \in V} 1 + \sum_{u \in V} \text{degree}(v)$$

$$= n + 2m \qquad \Theta(n+m)$$

Building a heap

60 14 15 35 12 11 26 23 30 17 40



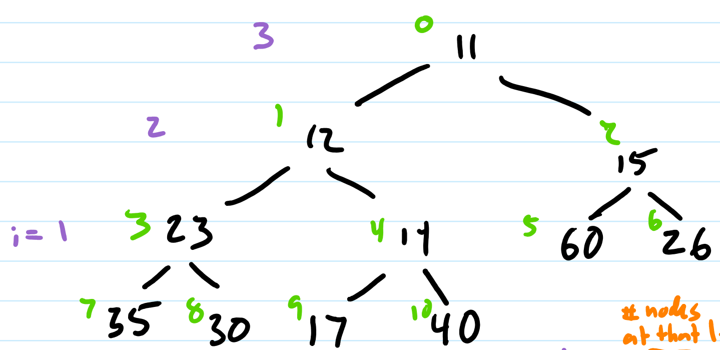
for each item
    enqueue(item, item)

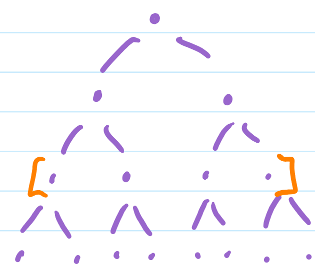
worst case
n iterations
$\Theta(\log n)$

$\Theta(n \log n)$

add all elements to an array
for i = last non-leaf downto root $\Theta(n)$ iterations
    reheap_down(i) $\le \log_2 n$

$O(n \log n)$





1 + 1 + 1

1 + 1   1 + 1

1 + 1 + 1 + 1

$\frac{n-1}{2} + \frac{n}{4} + \frac{n}{8} + \dots \approx n$

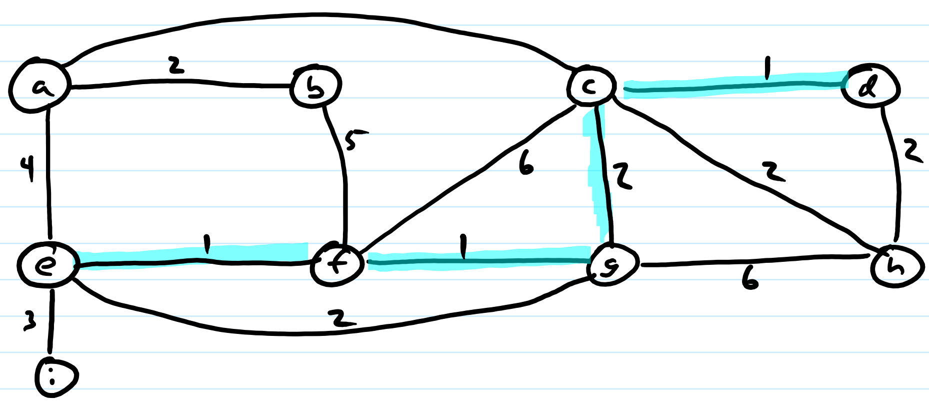
or total $\approx \underbrace{\sum_{i=1}^{\log_2 n}}_{\text{sum over all levels (counting from bottom)}} \overbrace{\frac{n}{2^i}}^{\text{\# nodes at that level}} \cdot \underbrace{i}_{\text{max swaps at level } i} = n \cdot \sum_{i=1}^{\log_2 n} \left(\frac{1}{2}\right)^i \cdot i \rightarrow \Theta(n)$

Disjoint Set Data Structure (UNION / FIND)

ADD (u) : add {u}

FIND-SET (u) : returns representative of set containing u
↳ same rep for any elt in same set

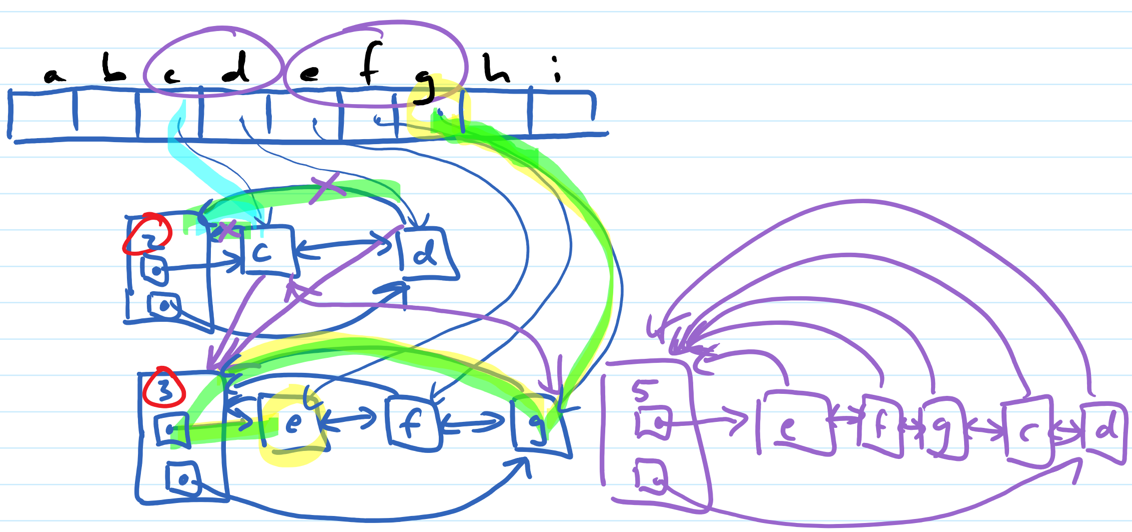
UNION (u, v) : unions set containing u w/ set containing v



0 : a
1 : b
2 : c
3 : d
4 : e
5 : f
6 : g
7 : h
8 : i

	a	b	c	d	e	f	g	h	i
rep	a	b	~~c~~	~~d~~	e	~~f~~	~~g~~	h	i
UNION(e,f)						e	e		
UNION(e,g)									
UNION (c,d)				c					
UNION (c,g)			e						

UNION(FIND-SET(c),
FIND-SET(g))



Let $R_{v,i} = \begin{cases} 1 & \text{if vertex } v \text{ relabelled during union } i \\ 0 & \text{otherwise} \end{cases}$

total # relabellings $= \underbrace{\sum_{i=1}^{n-1}}_{\text{sum over all } n-1 \text{ unions}} \underbrace{\sum_{v}}_{\text{sum over each vertex}} R_{v,i} =$

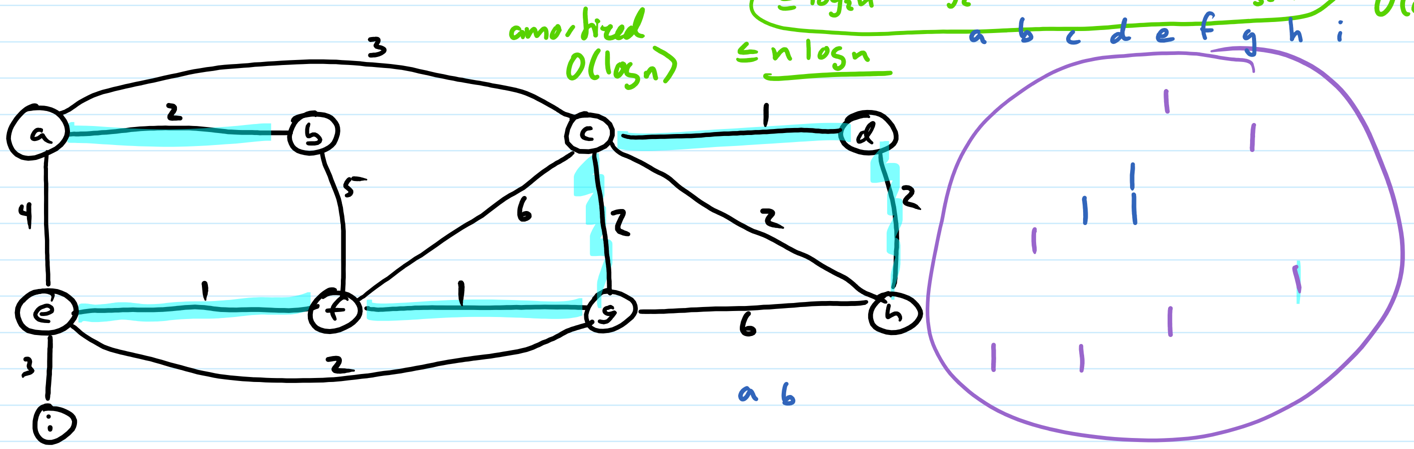
$\approx$ total work for all UNIONs

$$n-1 \left[\begin{array}{l} R_{a,1} + R_{b,1} + R_{c,1} + \cdots + R_{h,1} \le \frac{n}{2} \\ R_{a,2} + R_{b,2} + \quad - \quad - + R_{h,2} \le \frac{n}{2} \\ R_{a,3} + R_{b,3} + R_{c,3} + \cdots + R_{h,3} \le \frac{n}{2} \\ \quad \vdots \\ R_{a,n-1} + R_{b,n-1} + \cdots + R_{h,n-1} \le \frac{n}{2} \end{array} \right.$$

$\le \log_2 n \quad \le \log_2 n \quad \cdots \quad \le \log_2 n$ $\qquad \le (n-1)\frac{n}{2}$ $\quad O(n^2)$

a b c d e f g h i

amortized $O(\log_2 n)$ $\quad \le n \log n$

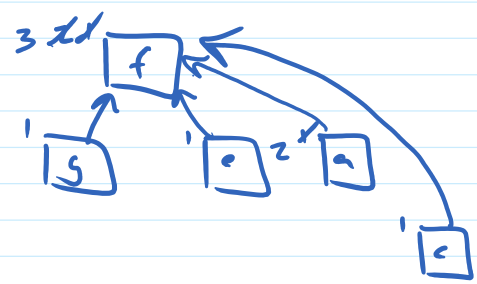


a b

f e g c d h

i

UNION by rank with path compression



UNION (a, c)
UNION (e, f)

find-set(c)

UNION (e, g)

UNION (a, c) $\quad O(\alpha(n))$ for each UNION/FIND-SET

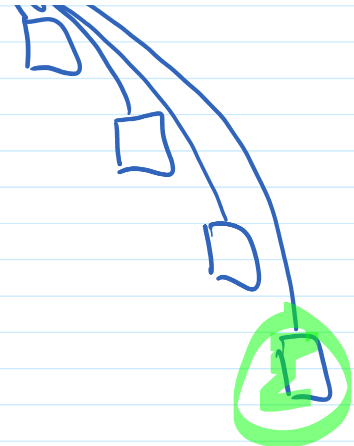
UNION (a, e)

$O(\alpha(n))$ for each UNION / FIND-SET
amortized

so $O(m \cdot \alpha(n))$ total for UNION/FIND part

still $O(m \log n)$

FIND-SET(g)



Ackerman's Function

$$A(0, n) = n+1$$
$$A(m, 0) = A(m-1, 1)$$
$$A(m, n) = A(m-1, A(m, n-1))$$

		n: 0	1	2	3	4	5
	0						
m	1						
	2						
	3						
	4						

Minimum Steiner Tree

Given weighted, undirected G and a subset of vertices A, find