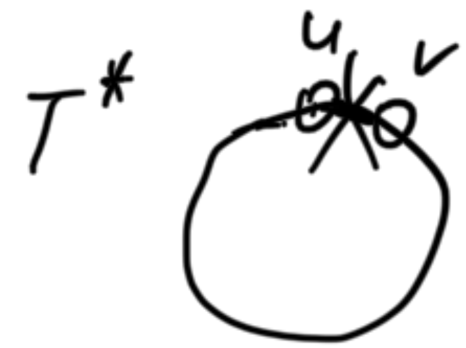
4.5 Minimum Spanning Tree (MST).

cost 3.

Given connected undirected $G=(V,E)$ with positive edge weights.

Goal: Find a set of edges $T^* \subseteq E$ s.t

- $G' = (V, T^*)$ is connected
- Minimize cost of $T^* = \sum_{e \in T^*} \ell(e)$



Solution T^* will be a tree.

pf: If T^* has a cycle C, assume $(u,v) \in C$

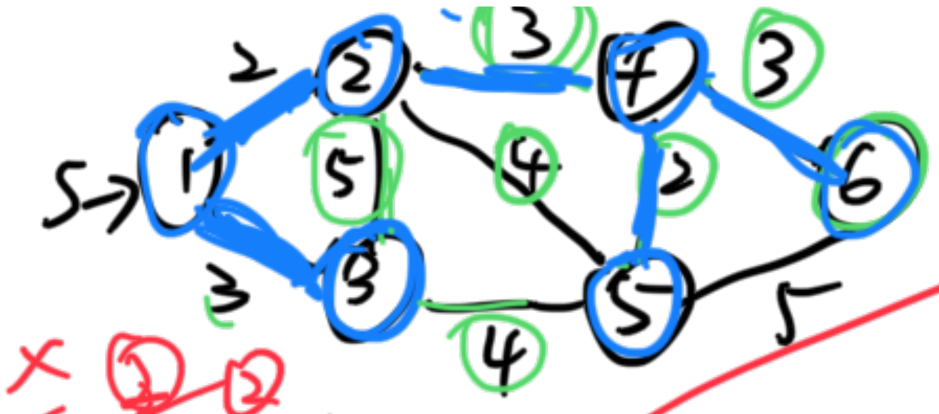
$\boxed{T^* - (u,v)}$, it will still be connected.

smaller cost connected subgraph ⇒ contradiction

Prim's alg: Maintain set X: nodes connected to source node s.

Initial $X = \{s\}$

$a[u]$: cost for adding u to X.



add u with smallest a[u] to X.

Initial $X = \{s\}$

$$\rightarrow \underline{a[u]} = \begin{cases} l(s,u) & \text{if } (s,u) \in E,\ p[u] = s \\ \infty & \text{otherwise} \end{cases}$$

	$X=\{1\}$	$X=\{1,2\}$	$X=\{1,2,3\}$	$X=\{1,2,3,4\}$
a[1]				
u{ 2}	2			
3	3	3		
4	∞ →	3	3	
5	∞ →	4 $p(5)=2$	4 →	2 $p(5)=4$
6	∞	∞	∞ →	3

for $i = 1$ to $n-1$

  $\boxed{u \leftarrow \text{node in } V-X \text{ with min } a[u]}$ ↑ pop

  Add u to X

  for v in neighbor(u):

    if $l(u,v) < a[v]$

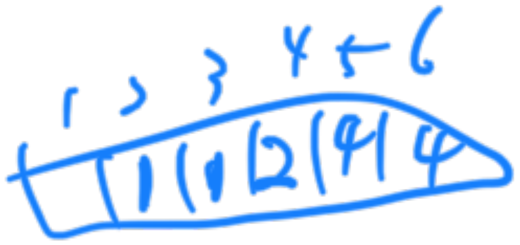
      $a[v] = l(u,v)$

      $p[v] = u$.

Dijkstra's alg: if $\boxed{d[u] + l(u,v)} < d(u)$

$d(v) = d(u) + l(u,v)$

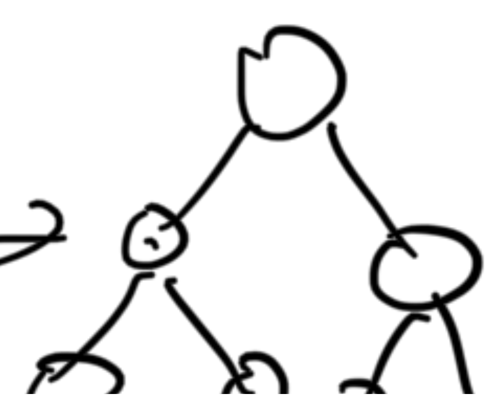
$(u, p[u])\ \forall u \rightarrow T^*$



Time Complexity: Use heap to store $a[u]\ \forall u$.

m update, n push. n pop.

$\Rightarrow O(m \log n + n \log n)$.

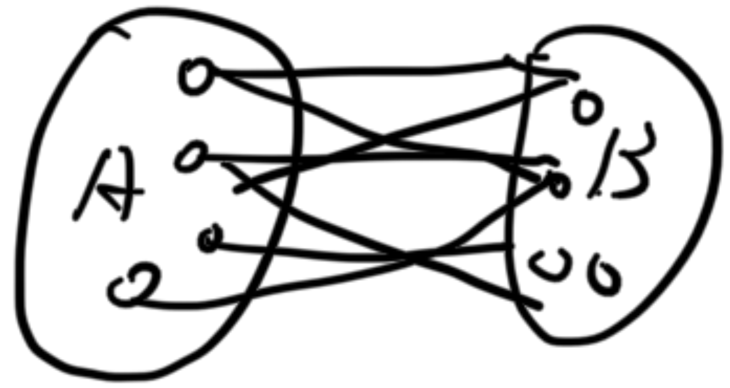


$\log m$

Correctness:

Define "Cut" of node sets A, B

$Cut(A,B)$: all edges between A, B



Prim's alg: each step X, $V-X$,

add min-cost edge in $Cut(X, V-X)$

to the MST.

Thm: If edge e is the minimum-cost edge in $Cut(X, V-X)$

for any X, then e must be in MST.

pf: If T^*: MST.

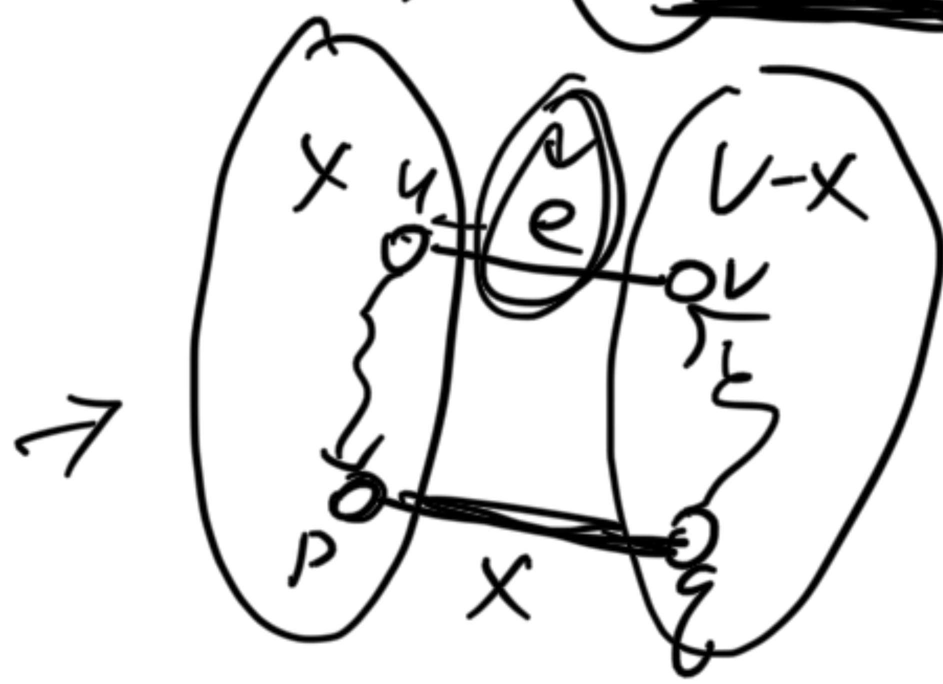
$e=(u,v)$ is min-cost edge in $Cut(X, V-X)$.

If e is not in T^*

If u, v connected by

$u \rightsquigarrow \cancel{p \to q} \rightsquigarrow v$

$T' = T^* - (p,q) + (u,v)$



T' is still a spanning tree.

$a \rightsquigarrow p \to q \rightsquigarrow b \quad T^*$

$\Downarrow$

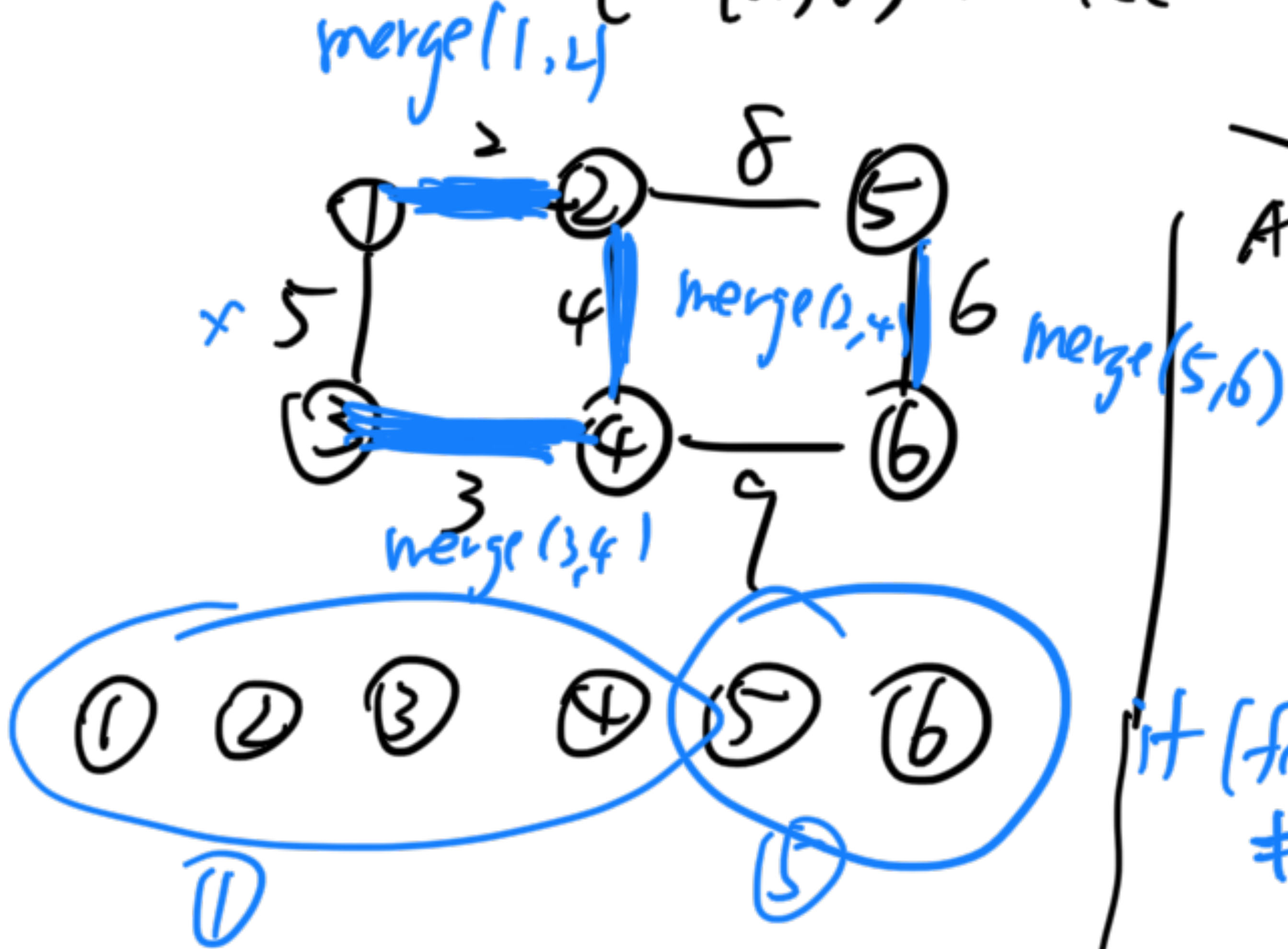
$a \rightsquigarrow p \rightsquigarrow u \to v \rightsquigarrow q \rightsquigarrow b \quad T'$

$\ell(T') < \ell(T^*)$ because $\ell(\underline{u,v}) < \ell(p, q)$. $\Rightarrow$ contradiction

Kruskal's alg:

- Sort edges from small to large.
- Consider edges one-by-one

$e = (u,v)$: add e if u v are in different connected components.



Alg: Initial $T = \emptyset$

- Sort edges s.t. $\ell(e_1) \le \ell(e_2) \le \dots$
- for $i = 1 \sim m$

  check $(u,v) = e_i$

  if (find(u) $\ne$ find(v)) $\to$ If u, v are in different connected component of T.

  $T = T + \{(u,v)\}$

  Union(u,v)

  stop if T already has $n-1$ edges.

Proof of correctness:

When adding $(u,v) = e$

u, v in different component.

$u \in X \quad v \in V-X$

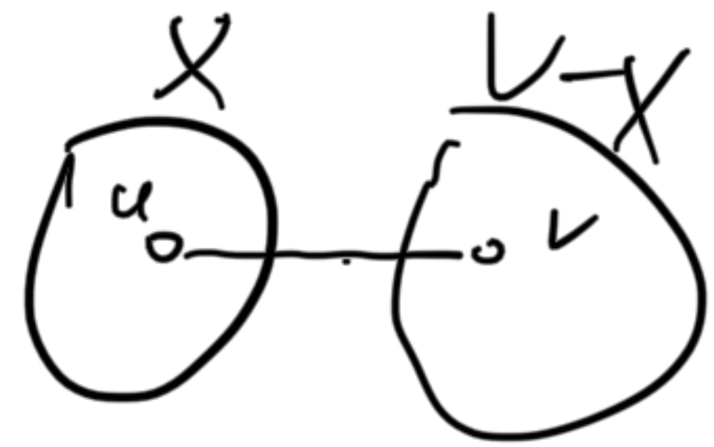
$e \in Cut(X, V-X)$

$\because$ $X, V-X$ are not connected,

e is the first edge in $Cut(X, V-X)$ being considered.

$\Rightarrow e$ is the min-cost edge in $Cut(X, V-X)$.

$\Rightarrow e$ is in MST



Union-Find : Store n elements and their sets.

- Initial : each element forms an individual set.
- Union(u,v) : merge the sets of u, v.
- Find(u) : return the "name" of the set.

: if $find(u) = find(v) \Leftrightarrow u, v$ are in the same set.

Tree-based Union Find :

n nodes,

a tree to denote a set

union(u,v) : merge two trees

union (1,2)

find (u) : return the root idx.

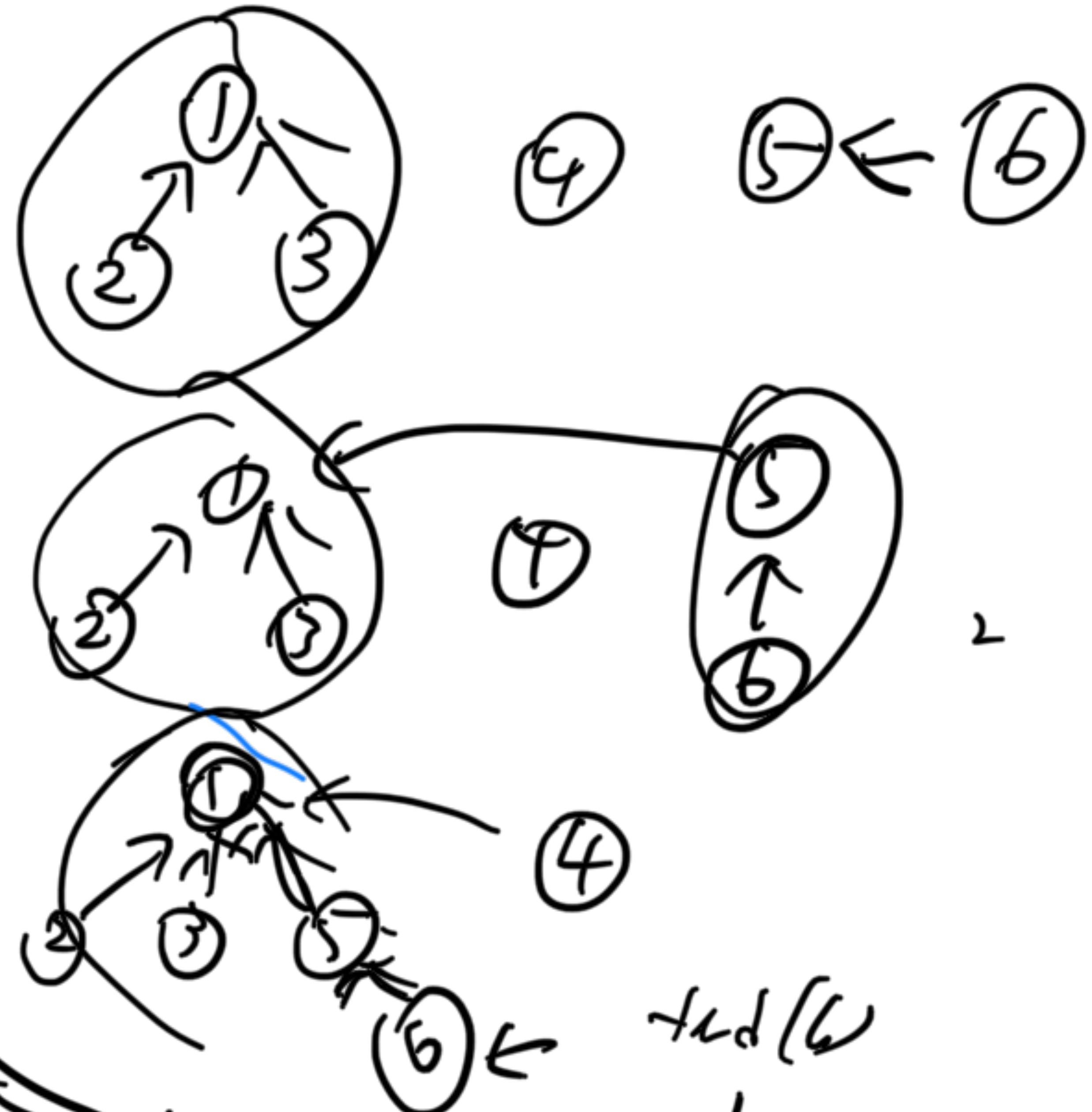
find(2) : 1

find(1) : 1

union(2,3) : find(2) → 1

find(3) → 3

depth = 2



union(5,6)

union(1,5)

find(6)

merge smaller tree to the larger one.

Cost of both union and find ≈ tree depth.

3

2

Claim: if the tree has depth $k \Rightarrow$ it has $\geq 2^k$ nodes

pf: tree depth k

$\Rightarrow \exists$ node $u \overset{k}{\rightsquigarrow}$ root

$\Rightarrow$ node u is merged to another set by k times

$\Rightarrow$ always merge with the bigger set

$\Rightarrow$ size $\geq 2^k$.

$\Rightarrow$ totally n nodes,

$\Rightarrow$ tree depth $\leq O(\log n)$.

$\Rightarrow$ both union and find cost $O(\log n)$ time.

$\geq 2C$