06.09.2024

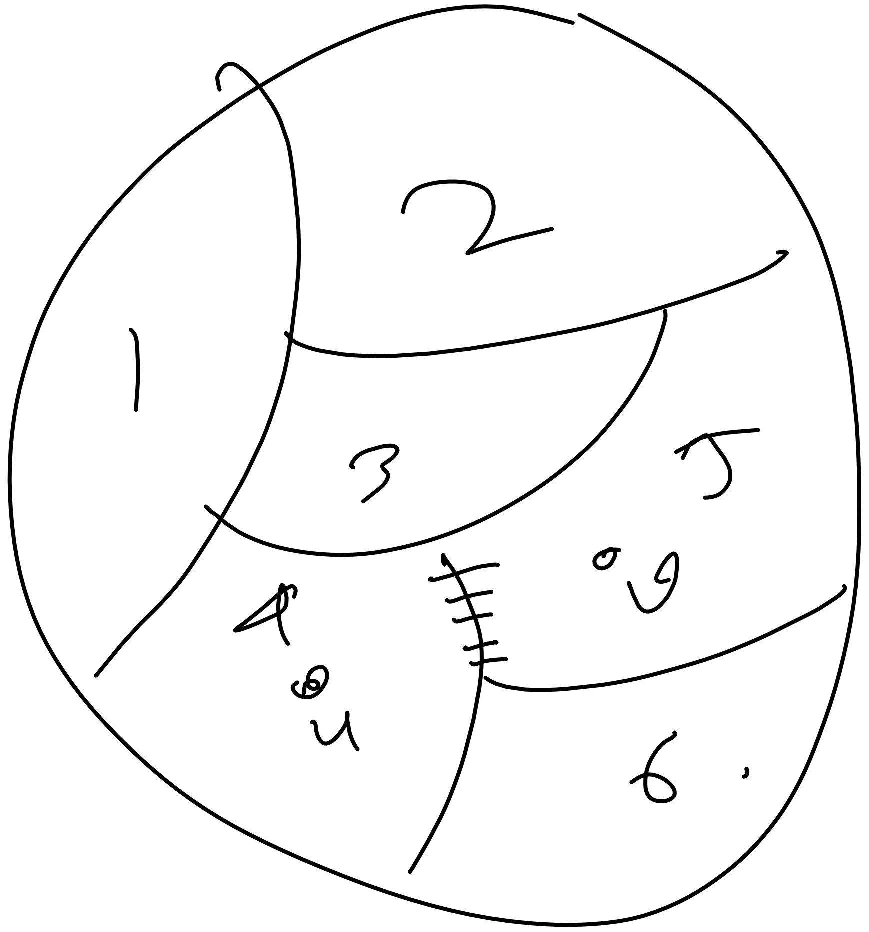
Equality.

$\boxed{\log n}$ — Find (u, v): yes iff $u, v \in$ same subset.

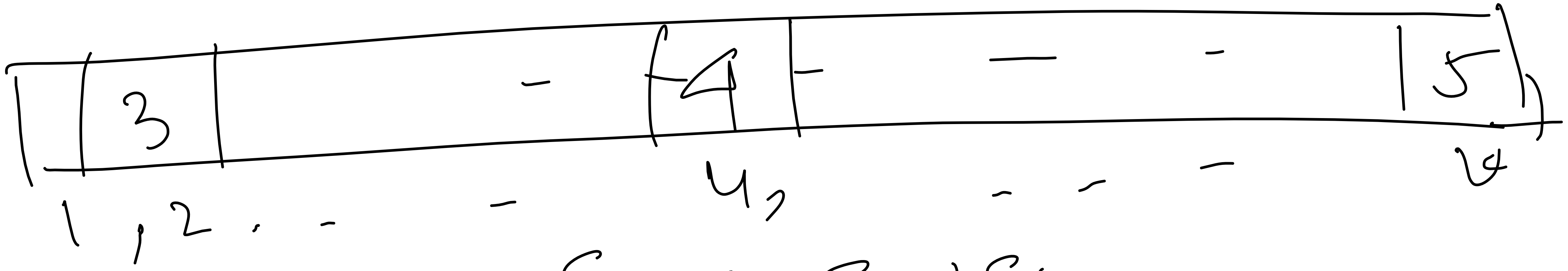
$\log n$

~~$O(1)$~~ — Merge(u, v): union of two subsets for u & v.

Disjoint Set / Union-Find.



U



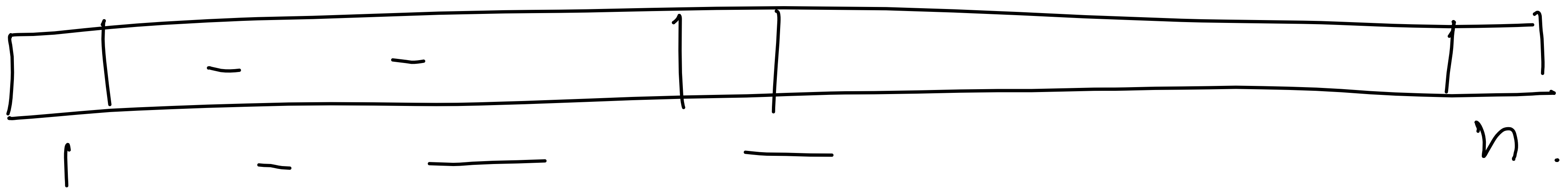
$S_i \leftarrow S_i \cup S_j$

$|U| = n$

Directed Trees

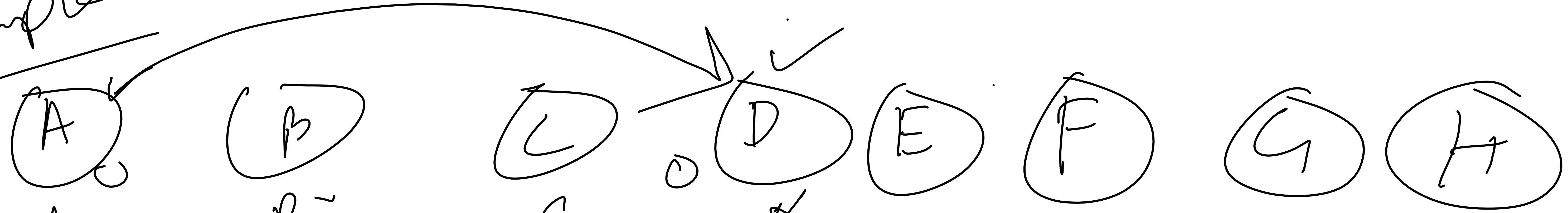
↓

parents ≤ 1

π: 

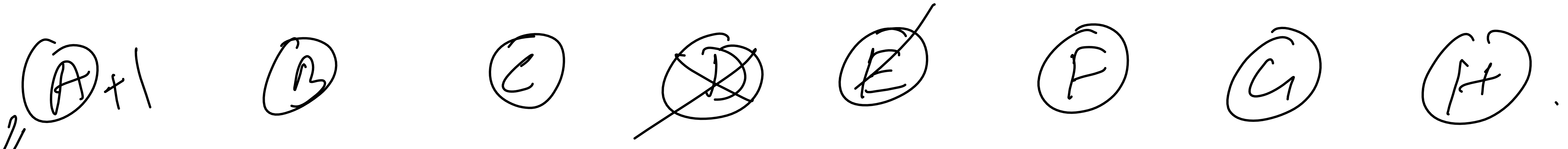
- We'll maintain each partition as a directed tree.
- Partition is identified with the root (source) of the tree.

Example:

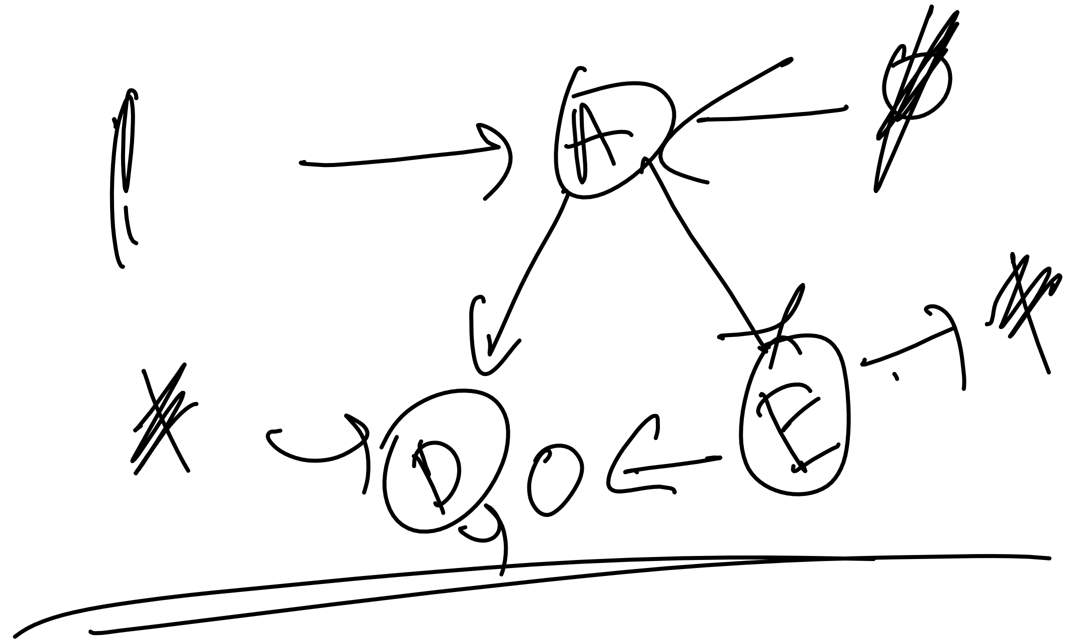


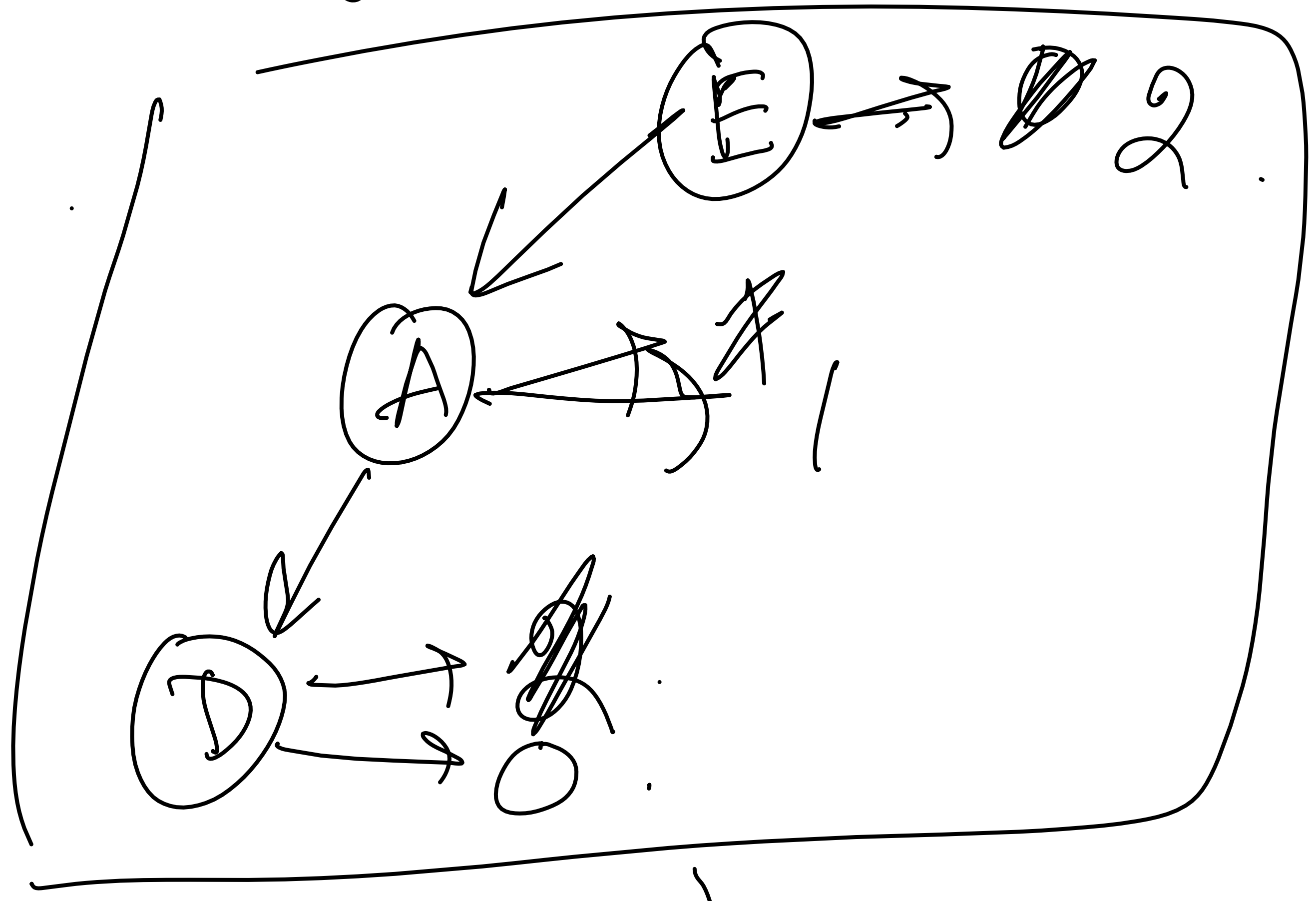
π: A B C ~~D~~ A ... H.

Merge (A, D).



Merge(E, D)





better!

worse!

$\therefore$ Shallow trees are better!

height:

	. — — —		0	—		
1	2 — — — — —					n

Initially every height is zero.

```
Procedure Find(u)
// returns the root of tree containing u.
temp ← u
While (π(temp) ≠ temp)
        temp ← π(temp)

return temp.
```

Time complexity: $\Theta(\log n)$.

Procedure Merge (u, v)

// Merge the two trees containing u & v

$\Theta(\log n)$.

$\Theta(\log n)$.

if $(\text{Find}(u) = \text{Find}(v))$

return.

~~height~~ $h_u \leftarrow \text{height}(\text{Find}(u))$

$h_v \leftarrow \text{height}(\text{Find}(v))$

$\Theta(1)$.

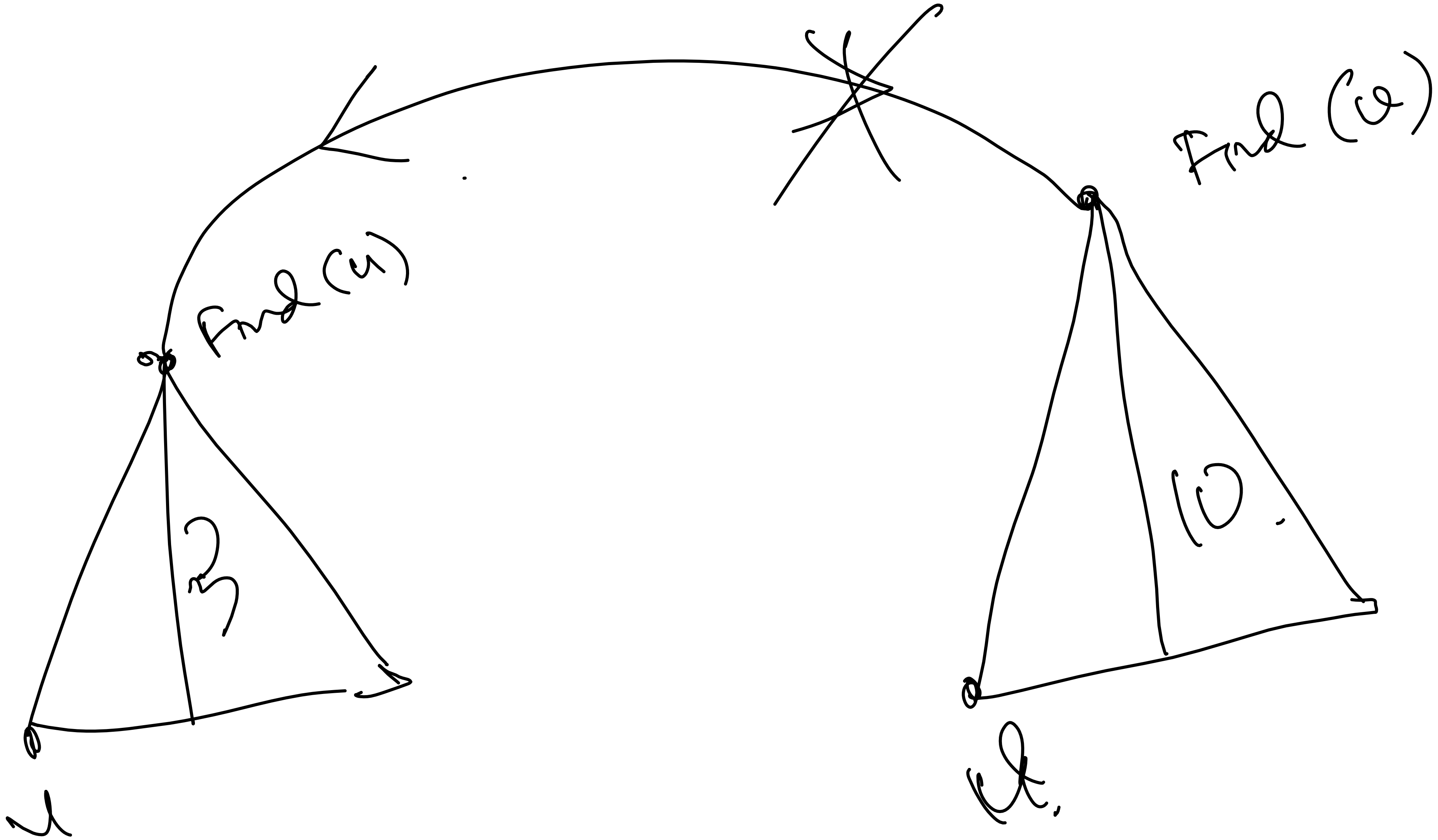
if $(h_u < h_v)$

$\pi(\text{Find}(u)) \leftarrow \text{Find}(v)$

else if $(h_u = h_v)$

$\pi(\text{Find}(u)) \leftarrow \text{Find}(v)$

height(Find(v)) ++

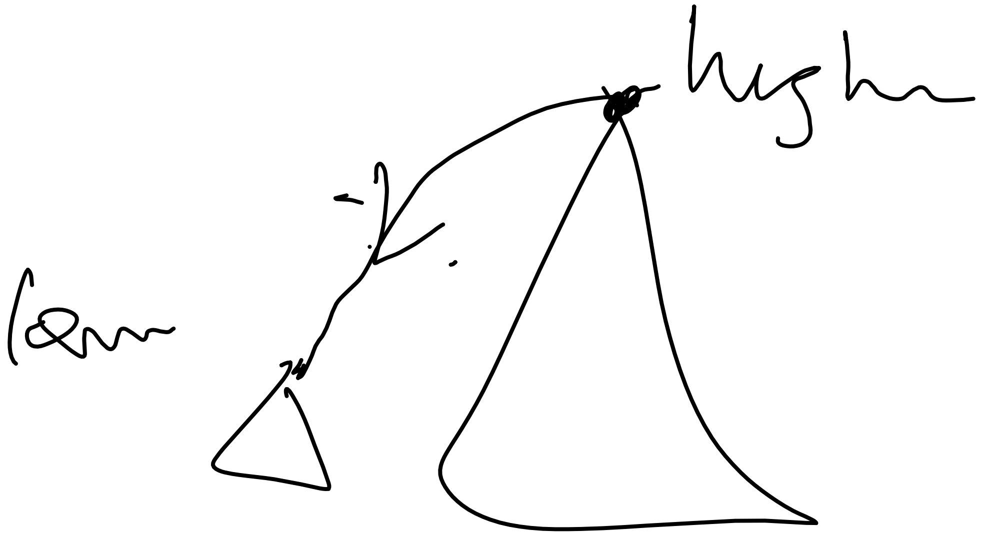


if ~~this~~ there is a

with

tree of height

are said.



Claim: If there are n nodes, the height of tree is $\Theta(\log n)$.

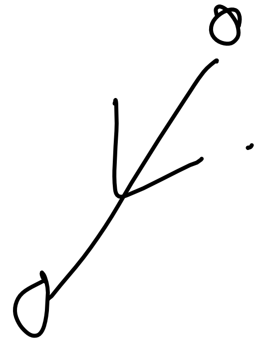
proof: Base: $n = 2$ node, height $= 1$.

I.H.: $\leq (n-1) \Rightarrow$ ~~$\Theta(\log(n-1))$~~ $c \cdot \log(n-1)$

$\geq 2^K$ nodes $\Leftarrow$ K.

I.S.

$\boxed{\text{I.H.:}}$ If the height of tree is K, it has got at least $c \cdot 2^K$ nodes for some c.



<u>base case:</u> $K = 1$

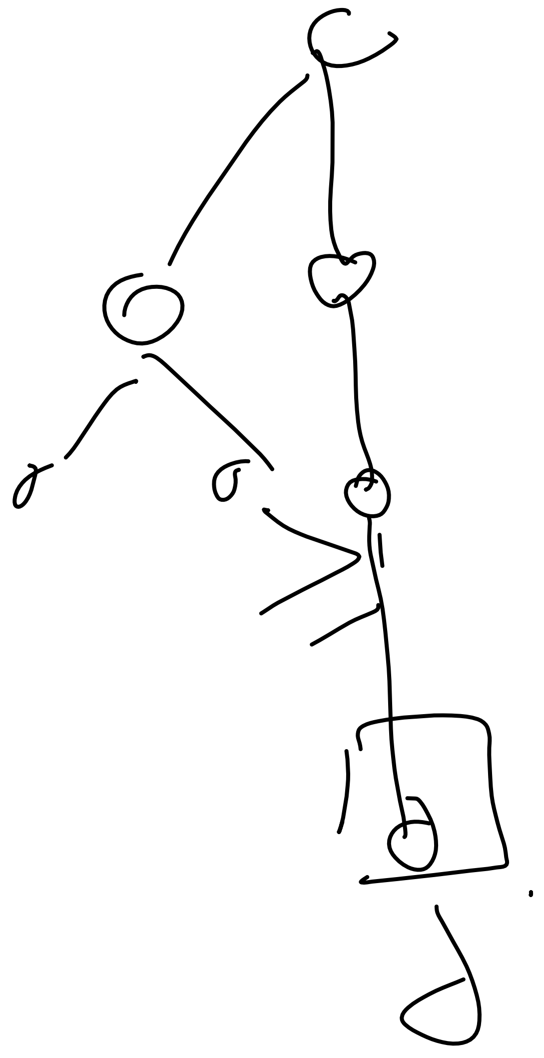
if height $= 1 \Rightarrow$ tree has ≥ 2 nodes.

<u>I.S</u>: Note the first time tree's height became $(K+1)$.

left had $\geq c \cdot 2^K$ nodes.

right side $\geq c \cdot 2^K$ nodes.

$\therefore$ the union tree $\geq c \cdot \underline{2^{K+1}}$ nodes.



$n = c \cdot 2^k$. $\quad k = O(\log_2 n)$

Time complexity of Merge function is $\Theta(\log_2 n)$.

Procedure MST (G)

$2 \to 1 \to w(2,1) \to$

// G = (V, E)

// G is given as weighted adjacency matrix.

F ← Sort list of edges from low weight to high weight.

$\Theta(|E| \log |V|)$

j ← 1

ctr ← 0

while (ctr ≠ n − 1)

(u, v) ← F(j).

$|E|$

if ~~(u,~~ (Find(u) = Find(v))

~~Conti:~~ j++

Continue.

log|V|.

else.

Merge(u, v).

ctr++

j++.

end if

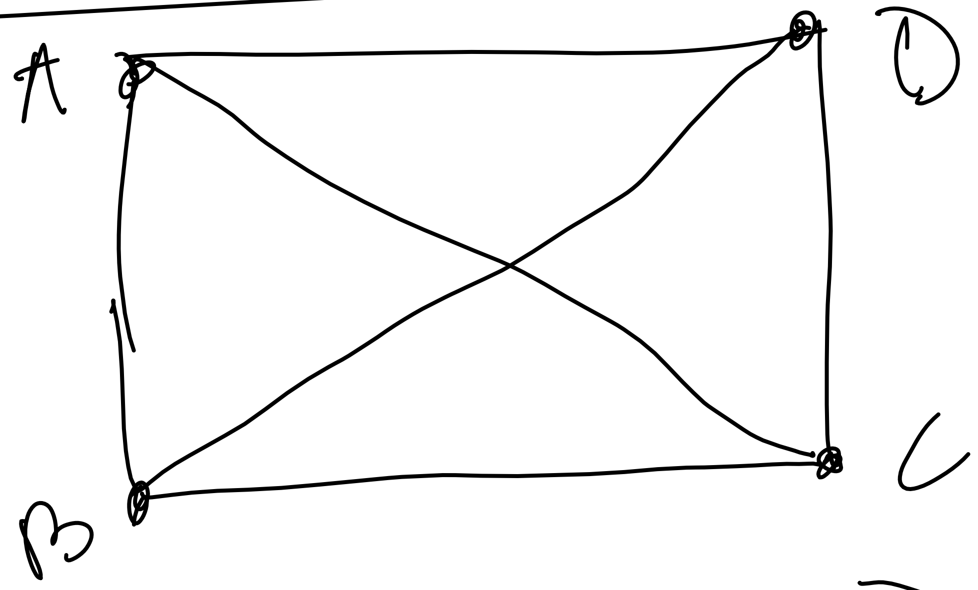
end while.

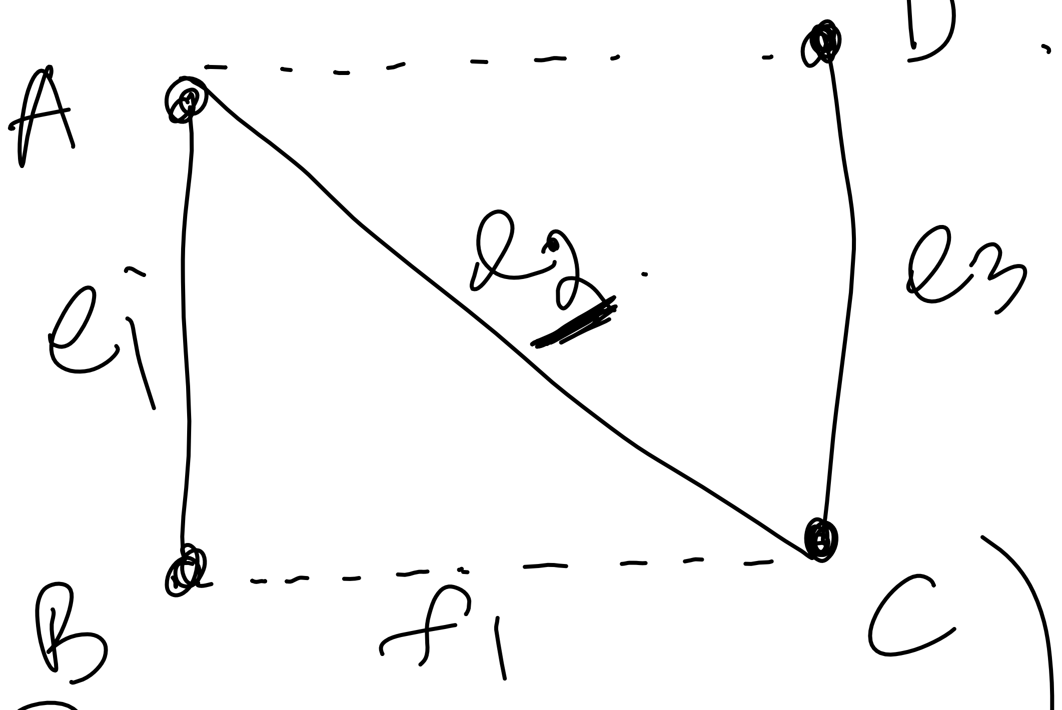
Kruskal's algorithm.

Time complexity: $O(|E| \lg |V|)$

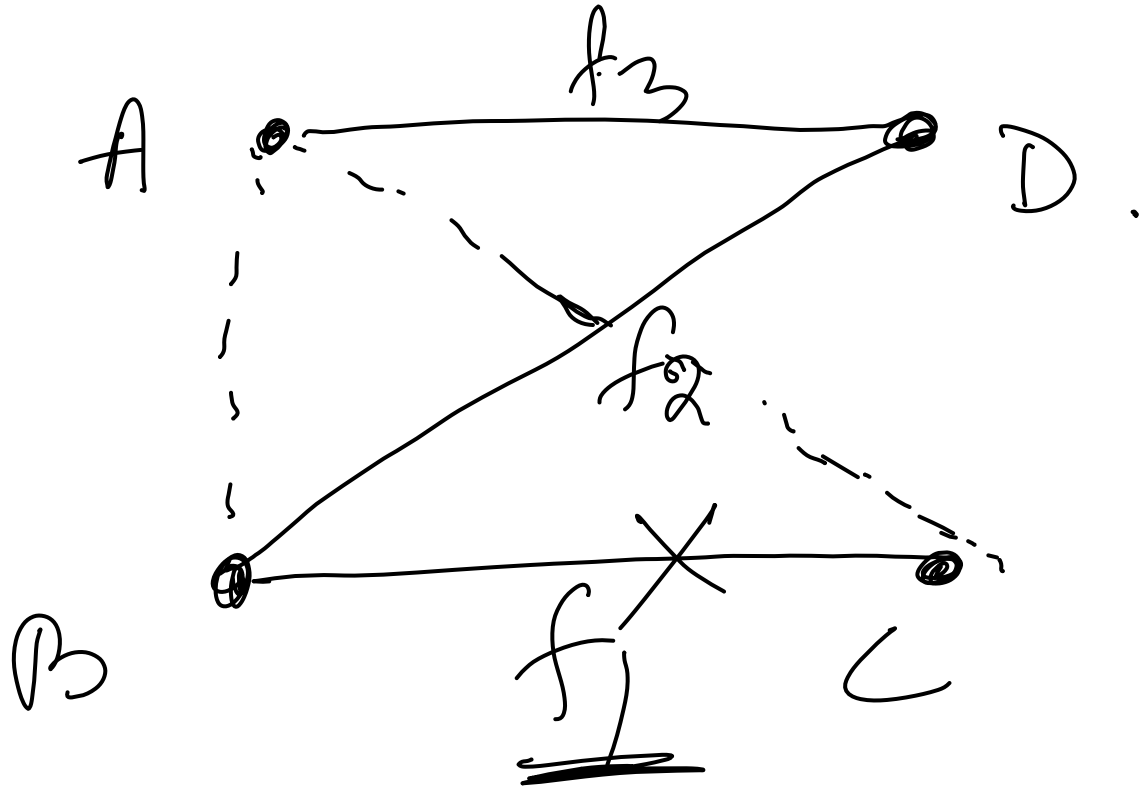
: Correctness :

Pairing Property of Spanning trees

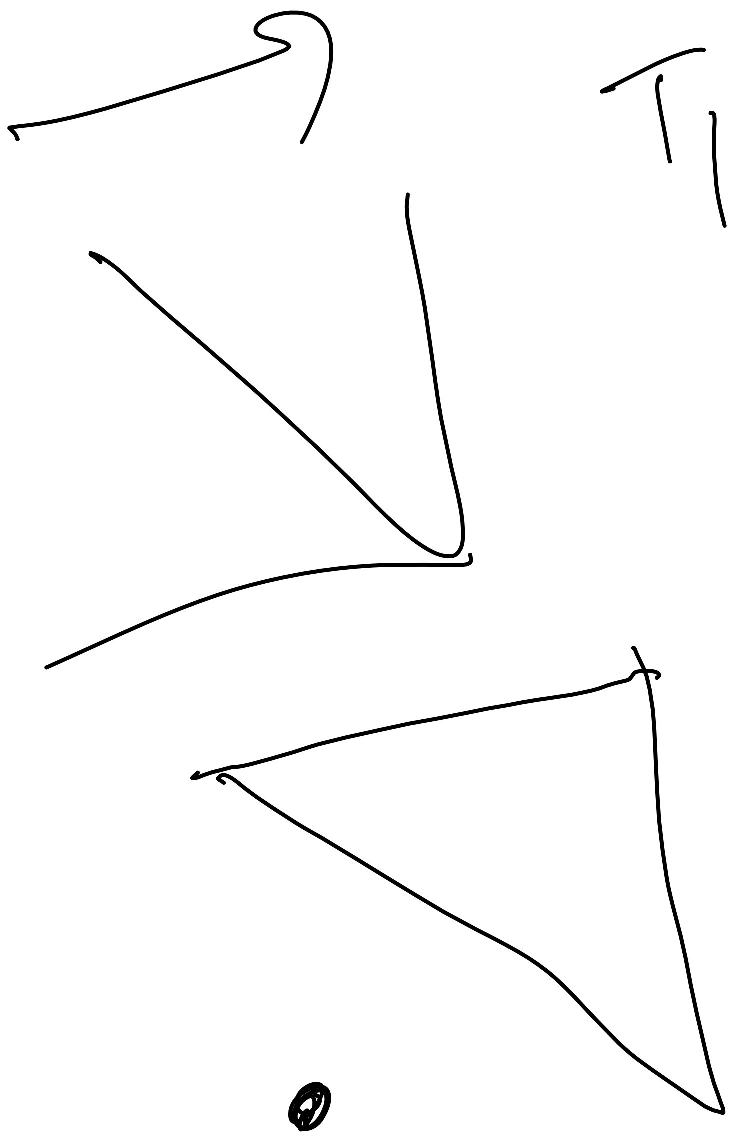




T_1



T_2



$T_1 \setminus T_2$	$T_2 \setminus T_1$
e_1	f_1
e_2	f_2
e_3	f_3

Pairing property:

$T_1 \cup \{f_i\} \setminus \{e_i\}$

is still a spanning tree.

06.09.2024

Lemma: For any two spanning tree T_1 & T_2, s.t. $T_1 \neq T_2$, there always exist a valid pairing.

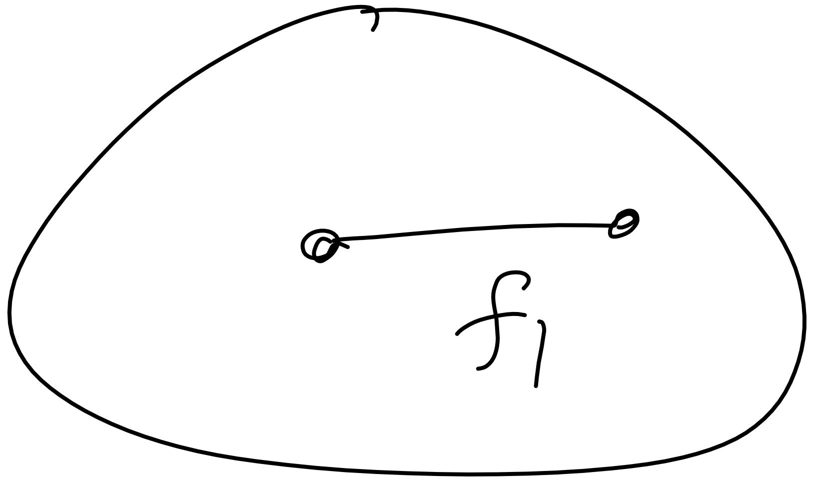
proof:

$$T_1 \setminus T_2 = \{e_1, \ldots, e_l\}.$$

$$T_2 \setminus T_1 = \{f_1, \ldots, f_l\}.$$

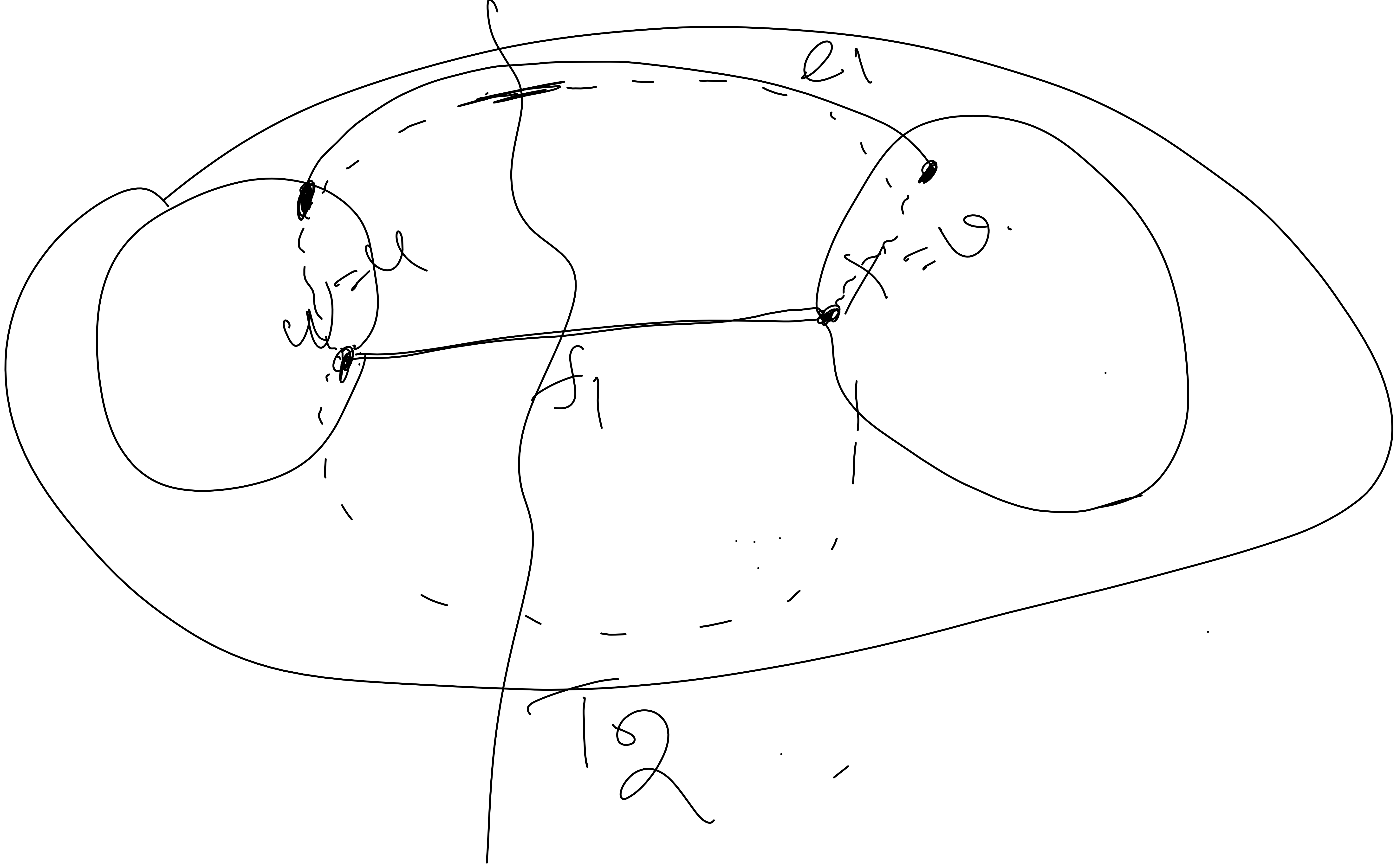
$T_1 \cup \{f_i\} \setminus \{e_i\}$ is a spanning tree $\forall i$.

T_1



T_2.

Pick some $f_1 \in T_2 \setminus T_1$



e1
u
w
v
x
f1
T2