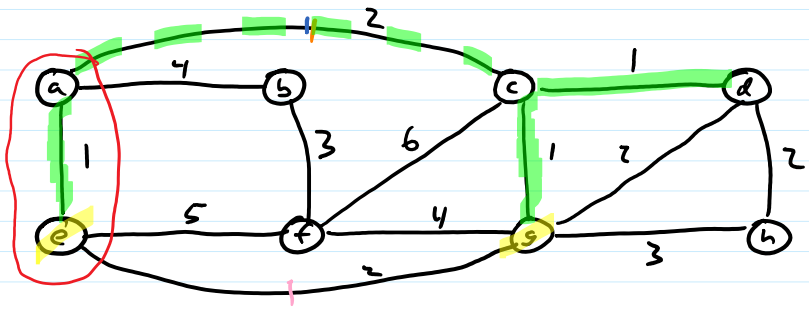
Kruskal's Algorithm: consider edges in order of ↑ weight
add edge if connects two different components of proto-MST

```
i ← 0
T ← ∅
sort edges in order of increasing weight
for each edge (u,v)
    if u,v in different connected components      FIND-SET(u) ≠ FIND-SET(v)
        T ← T ∪ {(u,v)}                          UNION(FIND-SET(u), FIND-SET(v))
    i ← i + 1
```

(u',v') in 1st i edges ⇒ u' —T→ v'

INVARIANT:
a) endpoints of 1st i edges are connected using edges in T
b) data structure corresponds to connected components in T
c) T ⊆ subset of 1st i edges
d) T is a proto-MST (subset of some MST)



(a,e) (c,d) (c,g) (d,g) (a,c) (e,g) (d,h) (b,f) (g,h) (a,b) (f,g) (c,f) (e,f)

connected components: {a} {b} {c} {d} {e} {f} {g} {h}
→ {c,d} {a,e}
→ {c,d,g}
→ {a,e,c,d,g}

Init: a) i=0, so says nothing b) singletons all 'round c) T=∅, i=0 d) T=∅ ⊆ any MST

Maintenance: Suppose a,b,c,d T before loop

a) if u,v aren't connected, go into if and connect them by adding (u,v) to T

b) every time we add to T, we call UNION

c) when we add (u,v), it is the $(i_{old}+1)^{st} = i_{new}^{th}$

d) Use Light Edge Theorem with cut comp(u), V-comp(u)

i) T is a proto-MST INV d

i) T is a proto-MST INV d

ii) T respects the cut

Let $(u',v') \in T$

(u',v') is in first i edges INV c

$u' \xrightarrow{T} v'$ INV a

Suppose (u',v') crosses cut and WLOG $u' \in comp(u)$, $v' \notin comp(u)$

$u \xrightarrow{T} u' \xrightarrow{T} v'$

$v' \in comp(u)$ (def connected comp), (alone)

⇒⇐

iii) (u,v) is light edge across cut ← $\forall (u',v')$, (u',v') crosses cut $\rightarrow c(u,v) \le c(u',v')$ if

$u \in comp(u)$, $v \notin comp(u)$

(u,v) crosses cut

Suppose (u',v') crosses cut

(u',v') is not in 1^{st} i edges

$c(u,v) \le c(u',v')$ (same reasoning as in ii) (edges sorted)

$\therefore T \cup \{(u,v)\}$ is a proto-MST

$= T_{new}$

Termination: At termination, all edges in G have been examined.

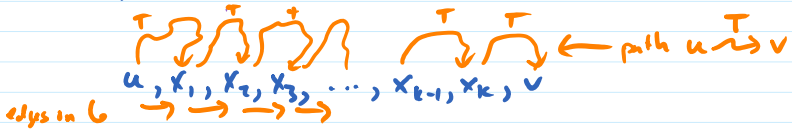
For any pair of vertices $u,v \in V$, (u,v) are connected in G (precondition: G connected)

[want: T connects all vertices]

there is a path $u = x_1, \ldots, x_k = v$ in G (def connected)

for all i, x_i, x_{i+1} are connected in T INV a

(u,v) are connected in T



$\therefore$ T spans G

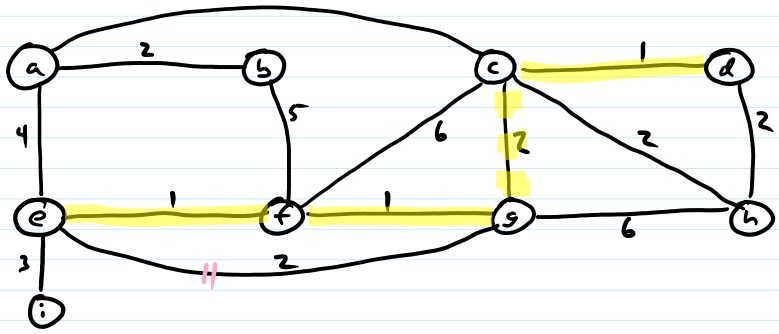
T is a proto-MST

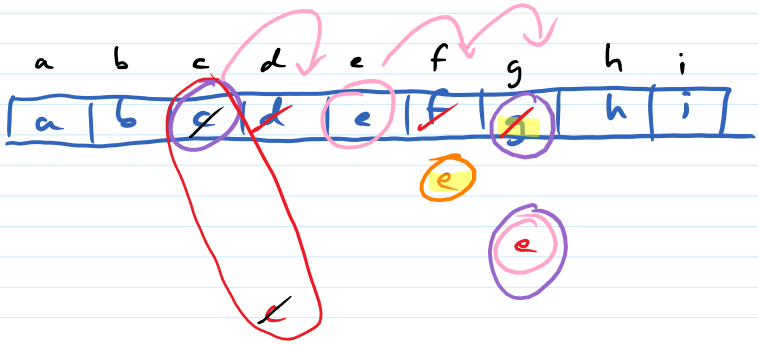
T is an MST

ADD (u) : add {u}

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

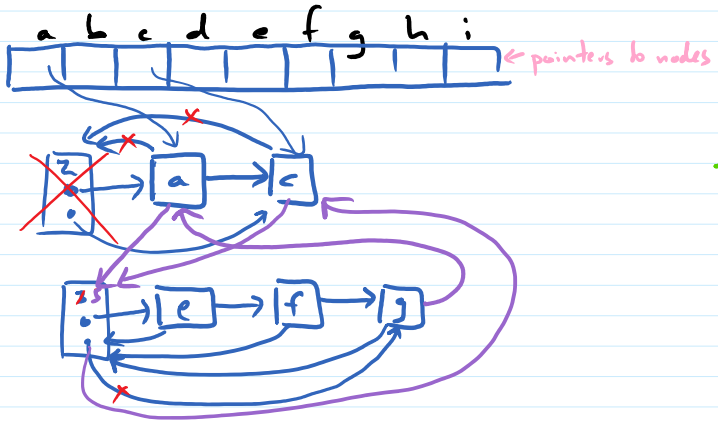
UNION (u,v) : unions set containing u w/ set containing v





UNION (e,f)
UNION (e,g)
UNION (c,d)
UNION (c,e)

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



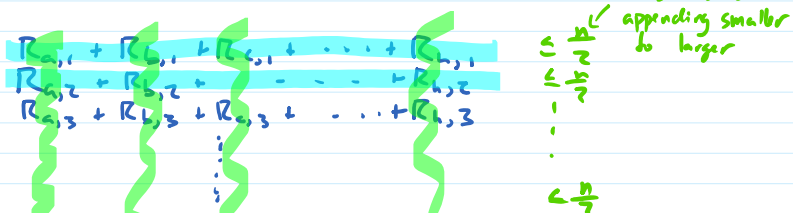
O(1) FIND-SET using pointers to 1st node

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

total # relabellings $= \sum_{i=1}^{n-1} \sum_{v} R_{v,i} =$ total work for all UNIONS

$\sum_{v}$: relabellings done by UNION #i

$\sum_{i=1}^{n-1}$: sum over all UNIONS



sum over all UNIONs

relabelings done by UNION #i

$R_{a,n-1} + R_{b,n-1} + \cdots + R_{k,n-1}$

$\le \log_2 n$ $\quad \le \log_2 n \quad \cdots \quad \le \log_2 n$

sum is $\le n \log_2 n$

$\le \frac{n}{2}$

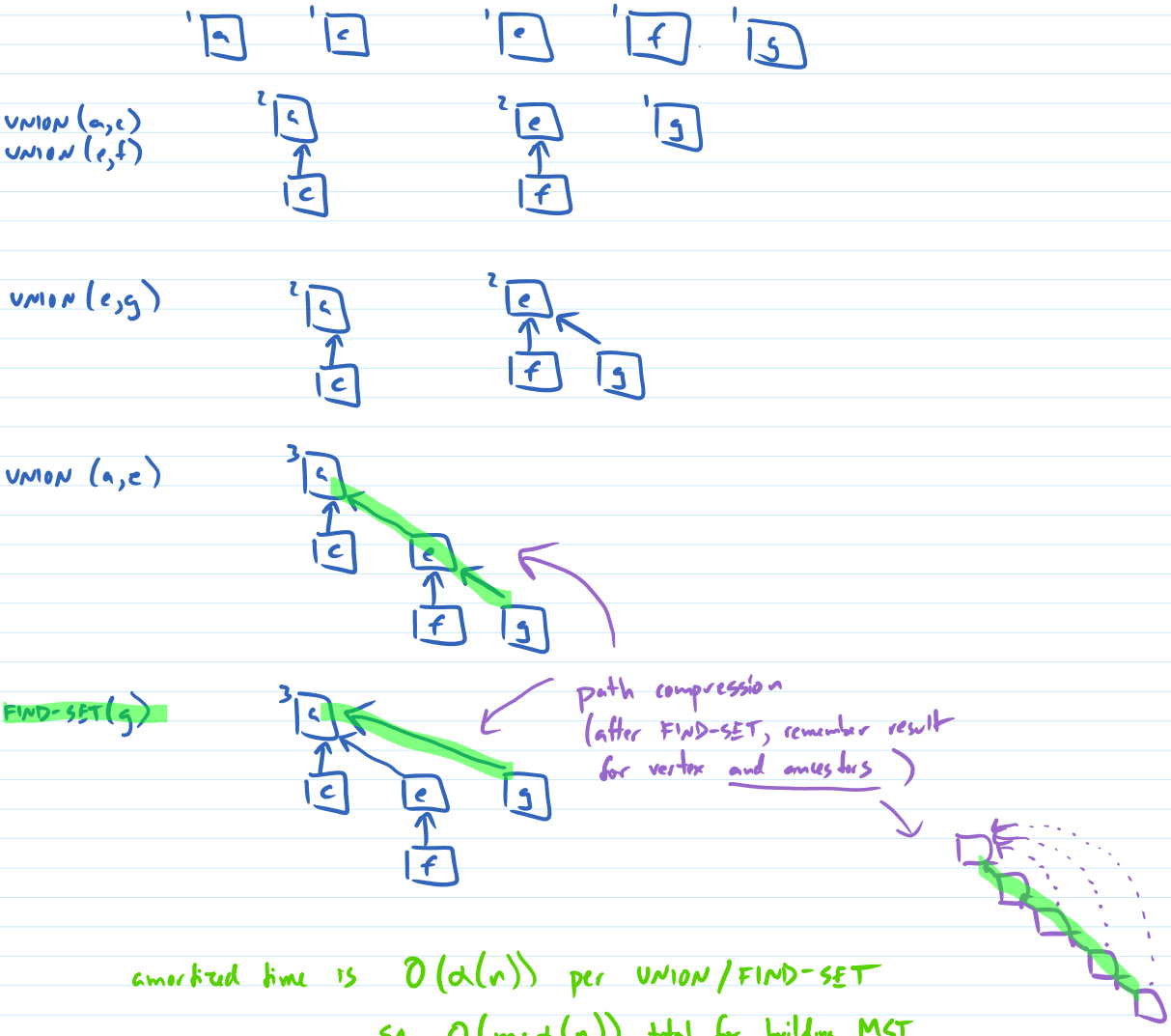
$(n-1) \cdot \frac{n}{2}$

overestimate b/c worst case can only happen once

total times vertex v is relabelled

$O(n \log n)$ total (so $O(\log n)$ amortized per UNION)

UNION by rank with path compression



amortized time is $O(\alpha(n))$ per UNION / FIND-SET

so $O(m \cdot \alpha(n))$ total for building MST

+ $O(m \log n)$ for sort $= O(m \log n)$ grand total

can do better in some cases if we know something about the weights (for example, integers in $0, \ldots, k-1$)