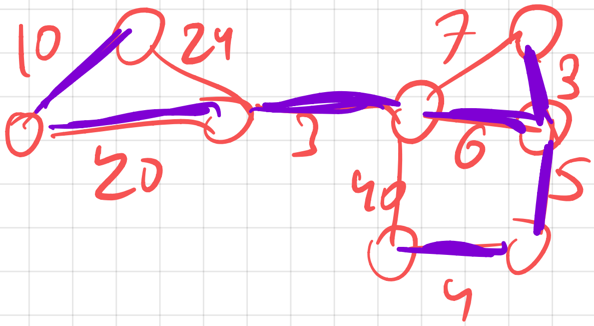
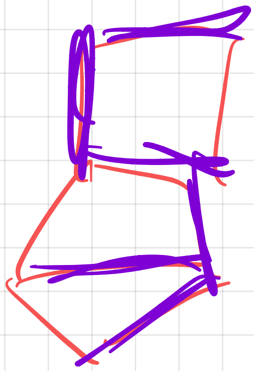
Minimum Spanning Trees



остовное (дерево)

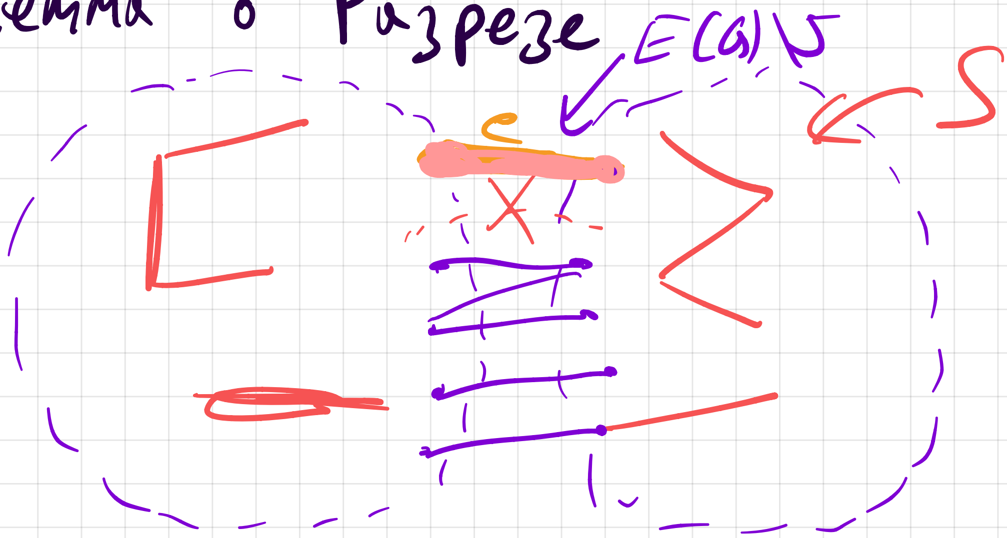


Минимальное остовное дерево:

- Краскала
- Прима

(*) Борувки

Лемма о Разрезе



$A \subseteq V(G)$ $\quad V(G) \setminus A$

Пусть $S \subseteq E(G)$ — такое мн-во рёбер, что $\exists T$ — мин. ост. дерево и $S \subseteq T$

Пусть $A \subseteq V(G)$, что ни одно ребро S не лежит на разрезе $[A; V(G) \setminus A]$

(иначе говоря $S \cap \delta(A) = \varnothing$)

Пусть $e \in E(G)$ и $e \in \delta(A)$

разрезу

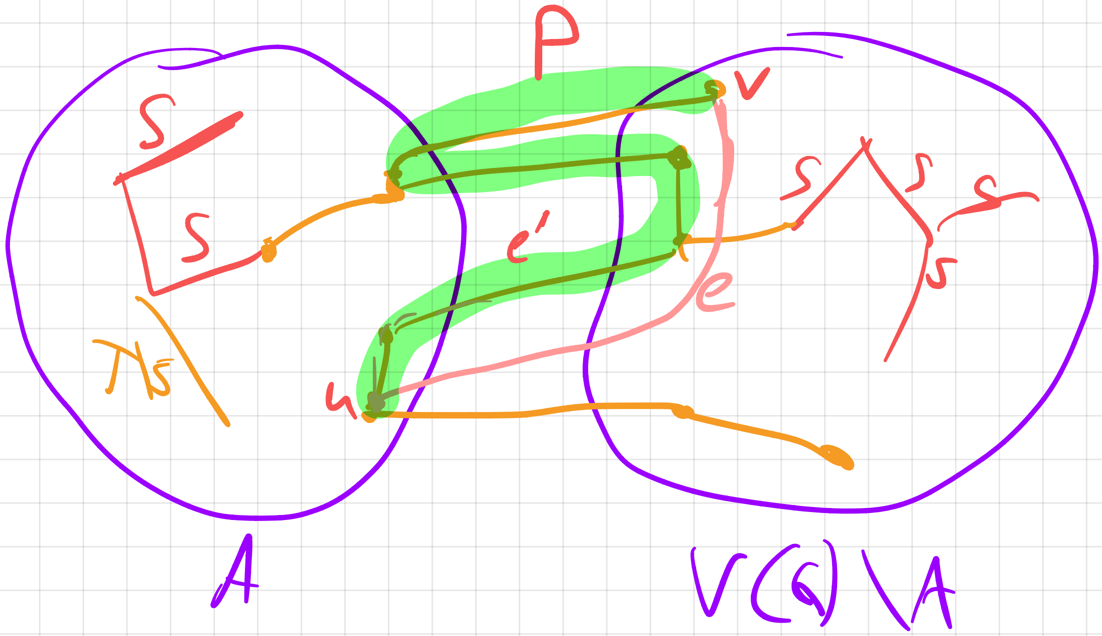
и w_e — мин. среди $[A; V(G) \setminus A]$
таких рёбер

Тогда e можно взять в ответ.

(т.е. $\exists T'$ — мин. ост. дерево и
$S \cup \{e\} \subseteq T'$)

<u>Д-во</u> (от. противного, т.е. пусть $\nexists T'$)

Рассмотрим T



$e \notin T \Rightarrow T+e$ содержит цикл.

Пусть $P+e$ — тот самый цикл.

Заметим, что $e \in \delta(A)$

тогда $P \cap \delta(A) \neq \varnothing$

Пусть $e' \in P$, это такое ребро

что $e' \in \delta(A$

Заметим, что $T' = T \setminus e' \cup \{e\}$ —

— ост. дерево

Заметим, что $S \cup \{e\} \subseteq T'$

Заметим, что $w(T') = w(T) - w_{e'} + w_e$

$\leqslant w(T) \Rightarrow$

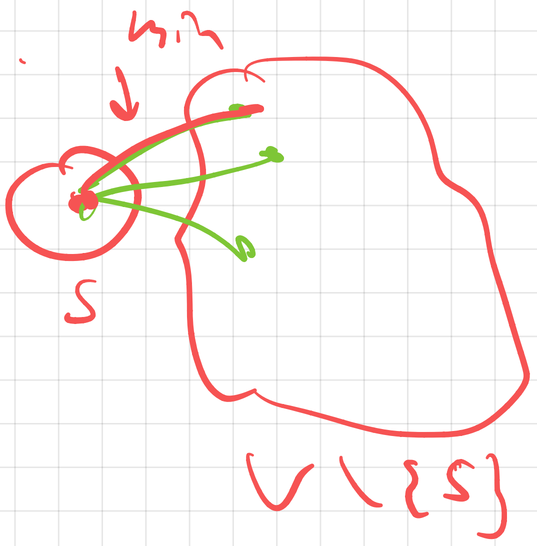
Т.к. T – M.S.T., то

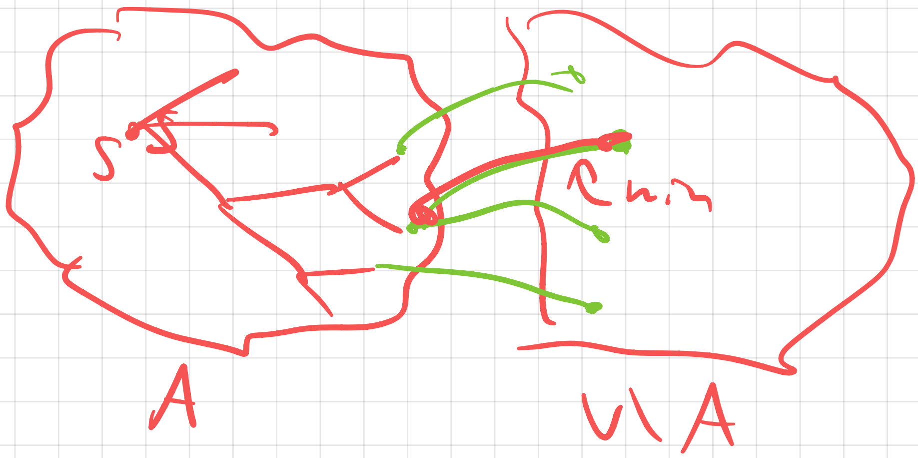
T' – тоже MST.

$\Rightarrow$ $S+e$ можно достроить до MST

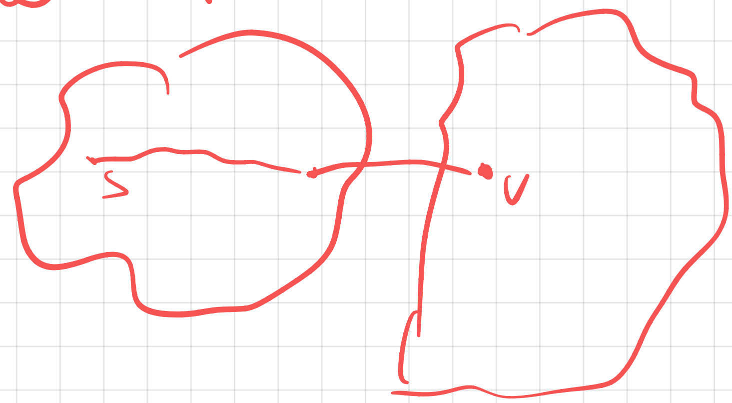
Алг. Прима

(похож на Дейкстра)



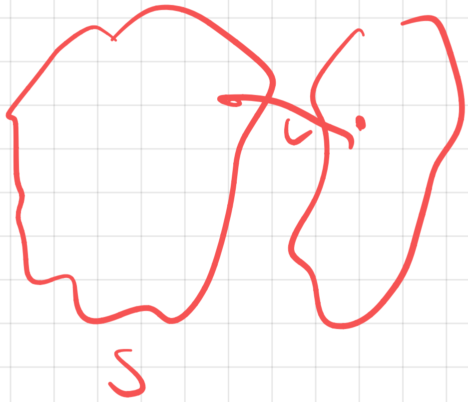


Дейкстра



d_v = min путь от старта проход. через разрез только посл. ребром

Прим



d_v = min ребро по которому можно подсоединить v в S

Прим:

$O(V^2+E)$ прост. реал.

$O(E\log V)$ binary heap

$O(V\log V+E)$ fibonacci heap / бр. куча алг. Краскал.

$Ans = \{\}$

for $e \in E$ в порядке сорт. по w_e

if $Ans + \{e\}$ не содержит цикл.

$Ans = Ans + \{e\}$

Алгоритм Краскала:

$Ans = \{\}$

while Ans не ост. дерево.

Пусть e: min ребро, что

$Ans + e$ ацикл

$Ans = Ans + e$.

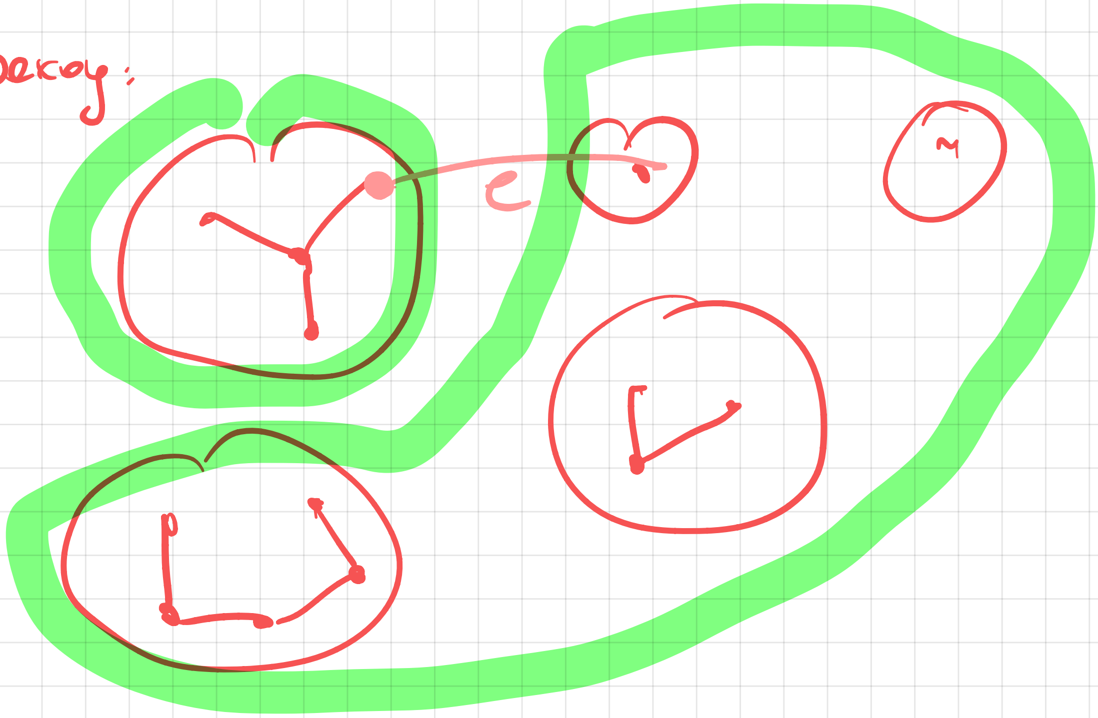
Утв: Ans подмн-во какого-то MST

в каждый момент времени.

Д-во. (по индукции)

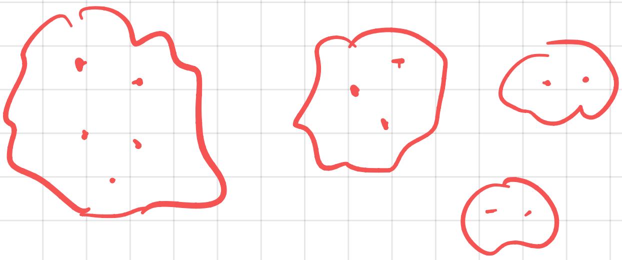
База: $Ans = \varnothing$

Переход:

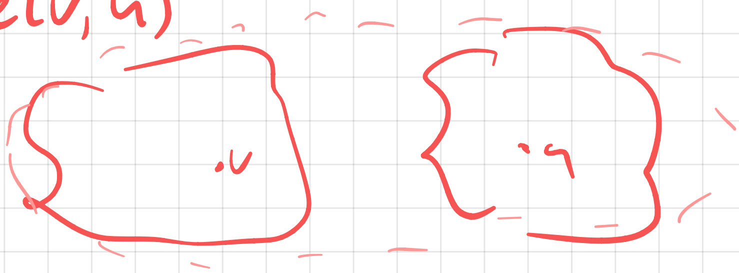


По лемме о разрезе.

Disjoint Set Union



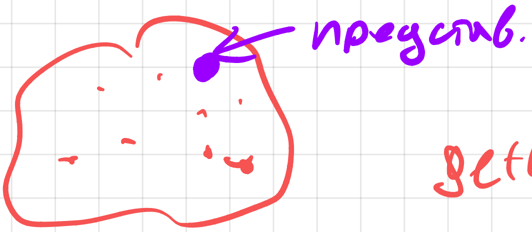
merge(v, u)



check (v, u)

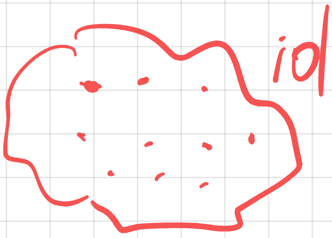
в одном множестве или нет?

get(v) по v получить представ. элемент.



$get(v) == get(u)$

Small to Large

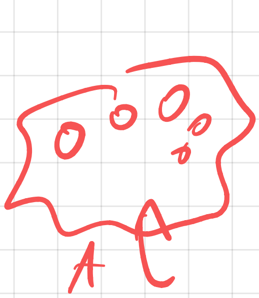


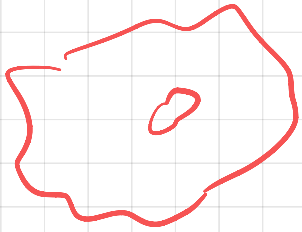
$lists[id]$ = список в-н компоненты #id

$comp[v] = id$ ← для каждой в-ны номер комп.

Check: $comp[v] == comp[u]$.

```
init {
  comp[v] = v
  lists[v] = {v}
}
```





```
merge(v, u):
    if comp[v] == comp[u]: return
    A = comp[v], B = comp[u]
    for w in lists[A]:
        comp[w] = B
        ~~lists[B] += {w}~~
    (lists[B].splice(lists[A]))
```

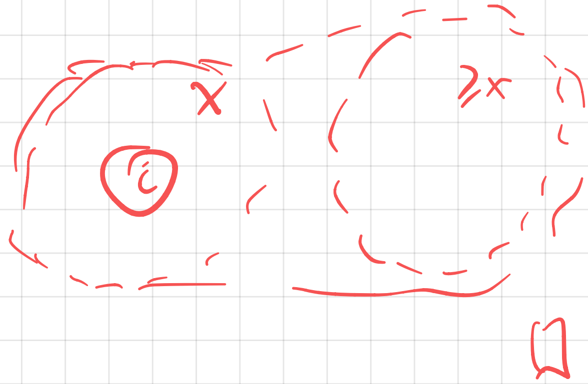
$O(|A|)$

добавьте перекр. меньший список в больший

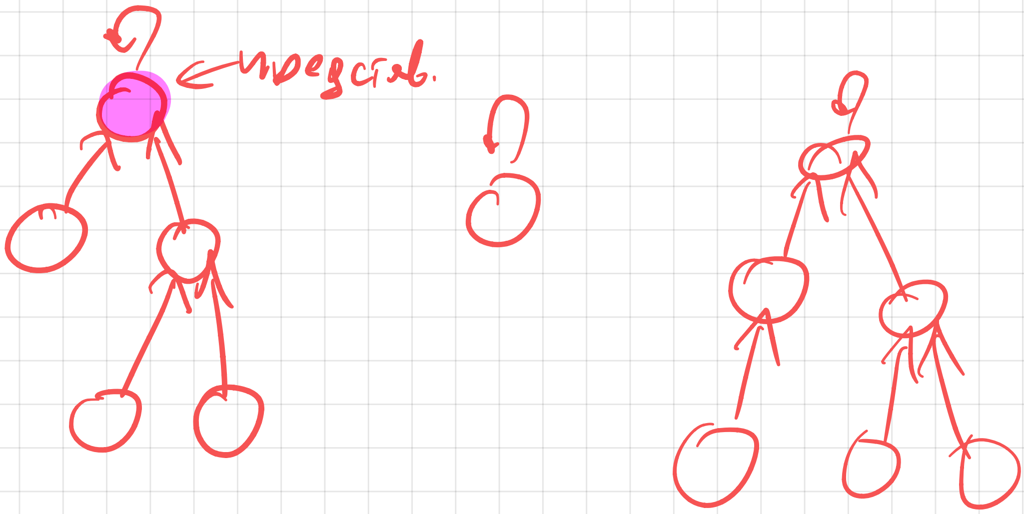
Утв: Суммарное время работы всех merge $O(n \log n)$

$\sum_{i \in [n]} k_i$ ← число перекр. эл-та i $\geqslant 2x$

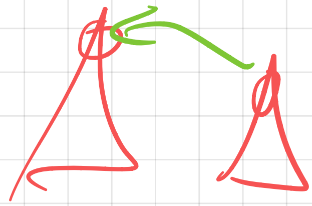
$k_i \leqslant \log n$



Древовидная Реализация



par[v] = v



```
get(v):
    if v == par[v]:
        return v
    else:
        return get(par[v])
```

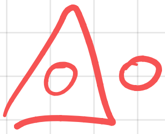
```
merge(v,u)
    v = get(v)
    u = get(u)
    if v == u: return false
    par[v] = u.
    return true
```

$T_{get} = \Theta(n)$ в худшем случае $\Rightarrow$:(

$T_{merge} = 2T_{get}$

$\Rightarrow$ время зависит от T_{get}.

ранги!

база 

merge: 

$= k$.

merge:

$= k+1$

(т.е. ранг. = max глуб. в поддереве)

(шаблон)

merge (v, u):

v = get(v)

u = get(u)

if (v == u): return

if !(rk[v] ≤ rk[u])

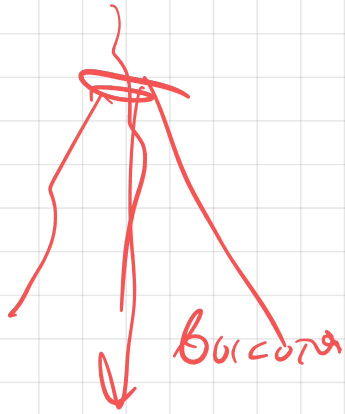
swap.

par[v] = u

if rk[v] == rk[u]:

++rk[u]

ранг. эвристика.



Утв: Если ранг. эврист., то $T_{get} = O(\log n)$

Д-во: $T_{get} = O(rk)$

Lm: размер поддерева у вер. с рангом k хотя бы $\geq 2^k$.

Д-во по инд-ии

$k=0$: база

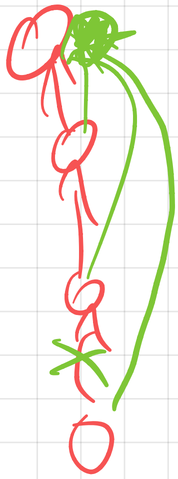
$\geq 2^0$

индукция

$\geq 2^{k-1}$ $\quad \geq 2^{k-1}$ $\Rightarrow$

$\Rightarrow \geq 2^k$.

Эвристика сжатия путей



```
get(v):
    if v == par[v]:
        return v
    else:
        par[v] = get(par[v])
        return par[v].
```

Утв 1 ранговая эвристика работает за $O(\log n)$ на Пут

Утв 2 Эвристика сжатия путей тоже работает за $O(\log n)$ в среднем

Утв $Э_{сж. путей} + Э_{ранг}$ работают за $O(\log^* n)$ в ср.

≤ 5

Утв $Э_{сж. путей} + Э_{ранг.}$ работают за $O(\alpha(n))$ в ср

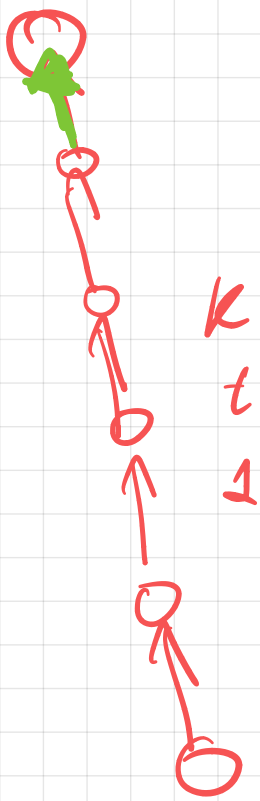
≤ 4

$$\left.\begin{array}{c} x \\ \downarrow \\ \log(x) \\ \downarrow \\ \log(\log(x)) \end{array}\right\} \log^*(x)$$

2^1 $\quad 2^2 = 4$ $\quad 2^{2^2} = 2^4$ $\quad 2^{2^{2^2}} = 2^{16}$ $\quad 2^{2^{2^{2^2}}} = 2^{65536}$

Утв 2. Пусть СНМ размера n и мы сделали m оп. get, тогда время работы $O((m+n)\log_2 n)$

$O(m\log_2 n + \underline{n\log n})$



k - лёгких рёбер ($\leq \log_2 n$)

t - тяжёлых рёбер ($\leq m\log_2 n$ в сумме)

1

$\to k+t+1$

Def: тяжёлое

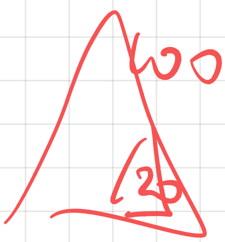
$sz(u) \geq \frac{1}{2} sz(v)$

$2\,sz(u) \geq sz(v)$

Def: лёгкое =

= $\neg$ тяжёлое.

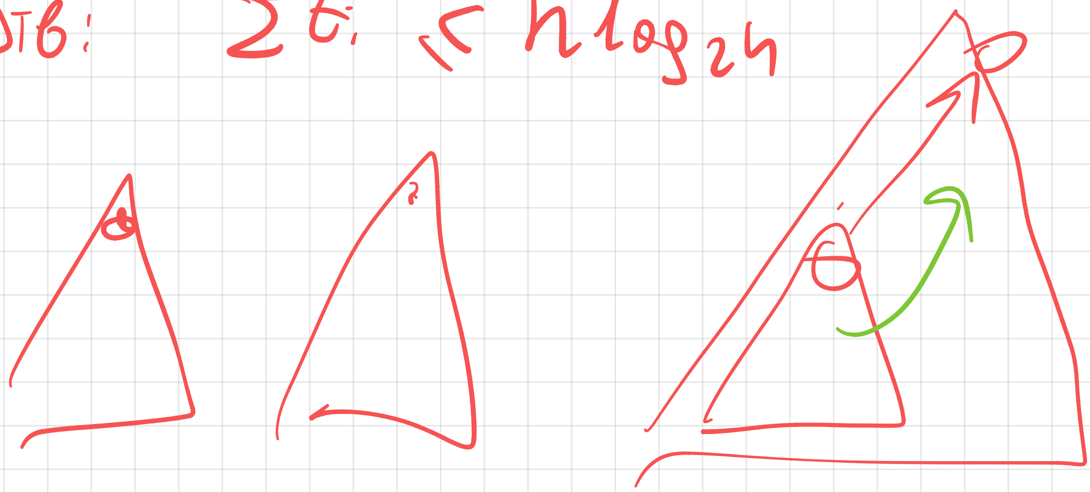
$sz(v) \geq 2\,sz(u)$



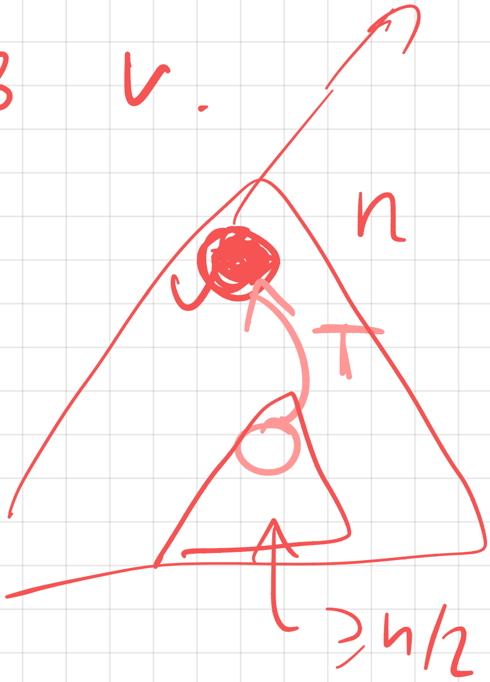
Утв: на пути наверх всегда $\leq \log n$ рёбер.

$\Rightarrow k \leq \log_2 n$.

Утв: $\sum t_i \leq n \log_2 n$



Утв: для каждой v будет $\leq \log_2 n$ проходов по T в v.



Заметим что после прохода по T размер $sz(v)$ уменьшается в ≥ 2 раза.

Также заметим, что т.к. v - не корень, то $sz(v)$ никогда больше не вырастет