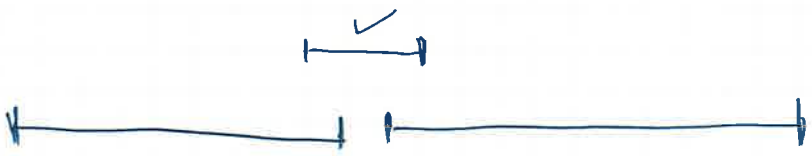
Optimal solution :

{4, 6, 7} ? No

{10, 11, 6, 7} ? Yes

{1, 11, 5, 7} ? Yes

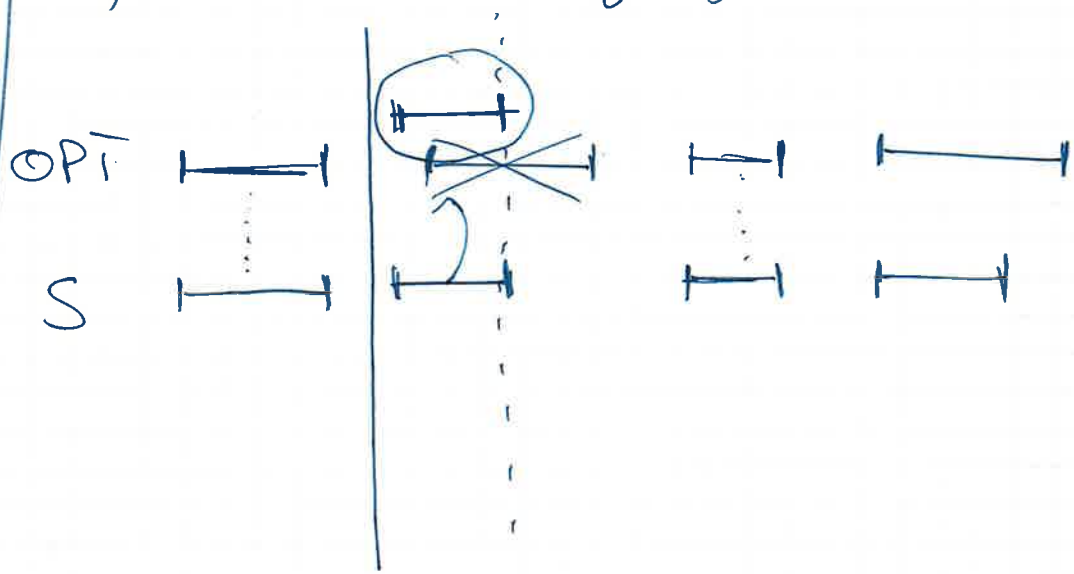
1) Increasing length



Not optimal

3) Increasing starting time

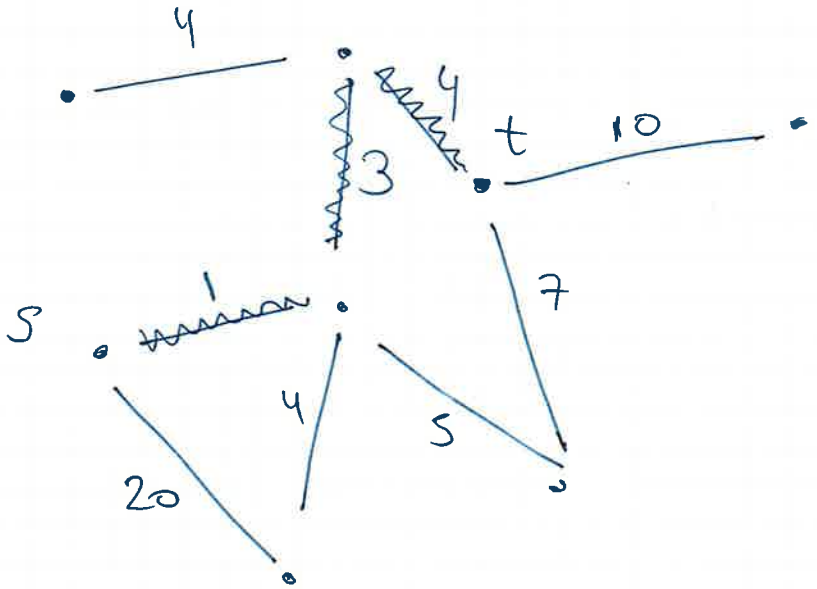
4) ~~Finish~~ Increasing f-time



⇒ New OPT is feasible

&

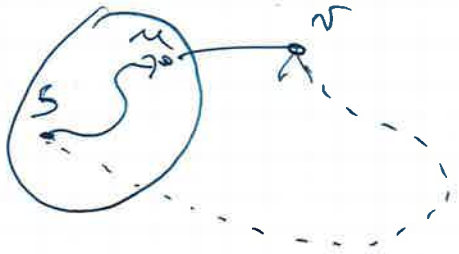
S and OPT agree on one more interval



Obs: At any point in time

$dist'(s,v) \geq dist(s,v)$

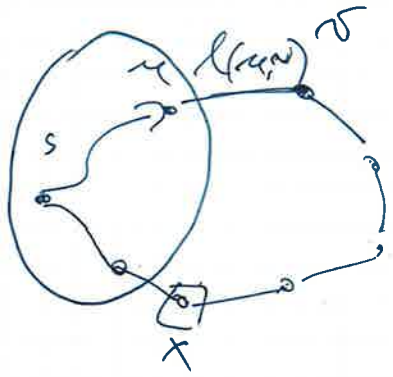
for all $v \in V \setminus S$



If v minimizes $dist'(s,v)$

then $dist'(s,v) = dist(s,v)$

Assume otherwise

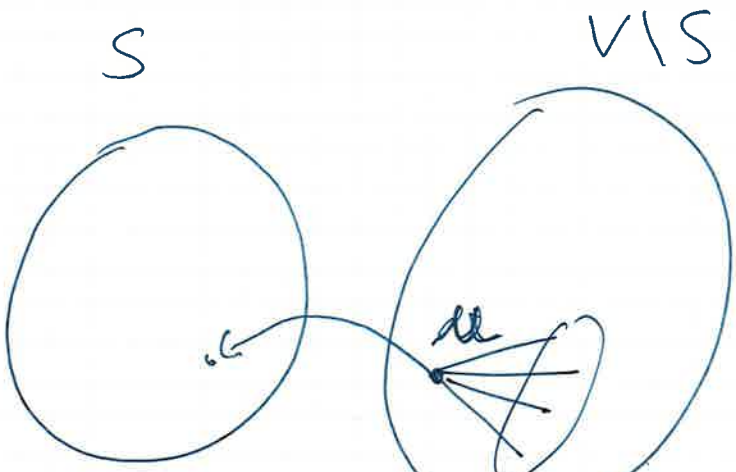


$dist(s,v) < dist'(s,v)$

$dist'(s,x) < dist(s,v)$

$\Rightarrow dist'(s,x) < dist'(s,v)$

A contradiction

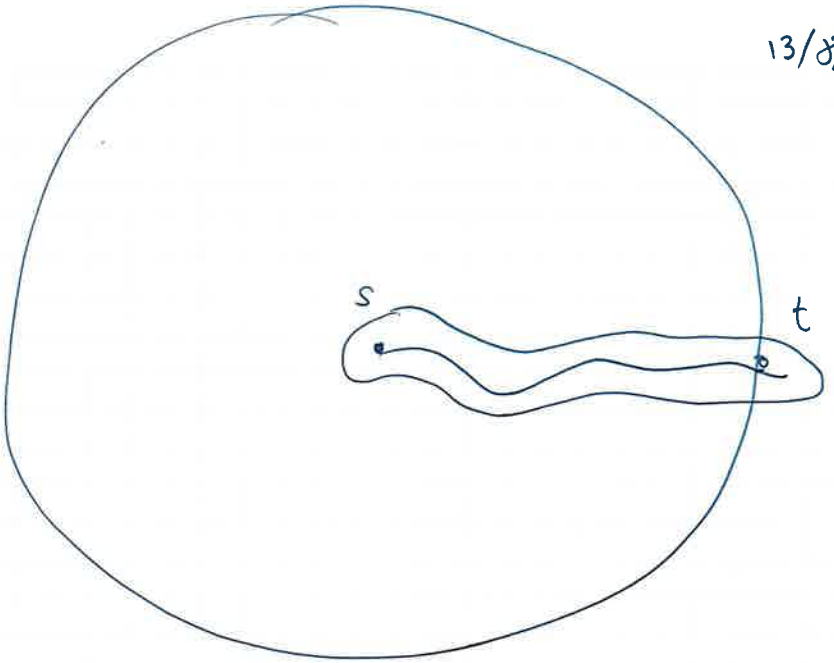


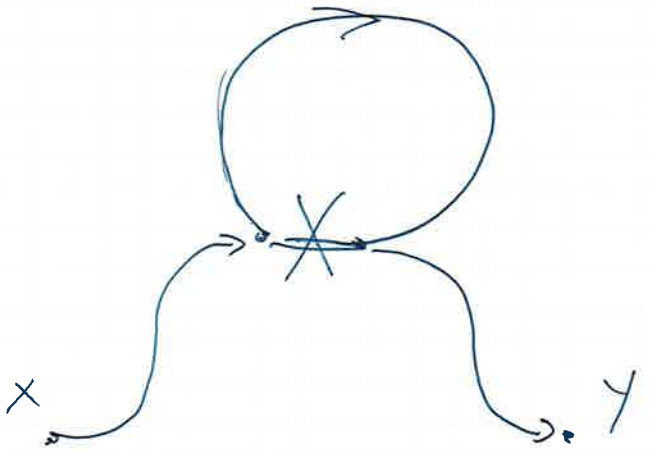
$\forall u \in S$:

$dist(s,u) = d[u]$

$\forall v \notin S$

$dist'(s,v) = d[v]$





Let (u,v) be edge with smallest cost

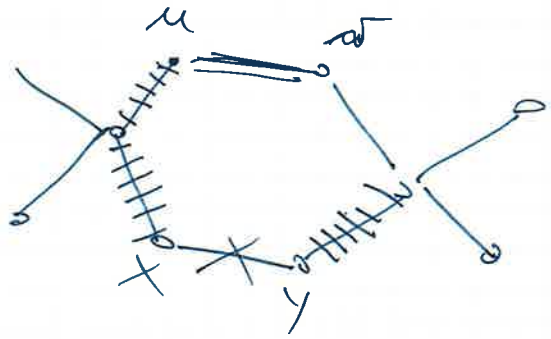
$\Rightarrow (u,v) \in OPT$

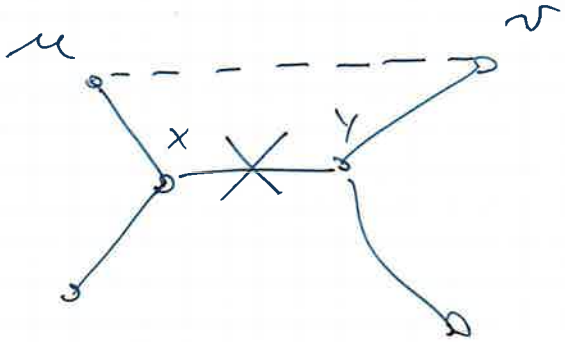
OPT & X

agree upto just before (u,v)

$(u,v) \in X$ & $(u,v) \notin OPT$

OPT →





take old tree

add (u,v)

delek (x,y)

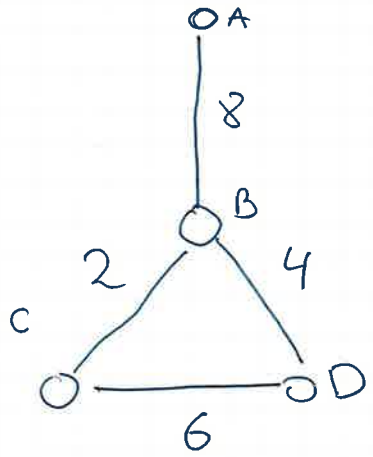
$\Rightarrow$ New tree has lower cost

 $\in OPT \cap X$

When (u,v) was added to X

want to avoid

$c(x,y) < c(u,v)$



Sorted in increasing weight

make-set(['A', 'B', 'C', 'D'])

find('C') ≠ find('B')

union('C', 'B')

{A} {B} {C} {D}

find('B') ≠ find('D')

union('B', 'D')

{A} {B, C} {D}

find('C') = find('D')

{A} {B, C, D}

find('A') ≠ find('B')

union('A', 'B')

{A, B, C, D}

$V = \{0, 1, \ldots, n-1\}$

$id = [0, 1, 2, 3, \ldots, n-1]$

Initially $id[v] == v$

	[0, 1, 2, 3, 4]
union(0, 1)	[1, 1, 2, 3, 4]
union(2, 3)	[1, 1, 3, 3, 4]
union(2, 4)	[1, 1, 3, 3, 4]
union(0, 3)	[4, 4, 4, 4, 4]

---

update id of first set

union (0, 1)
union (0, 2)
union (0, 3)
.
.
.
union (0, n-1)