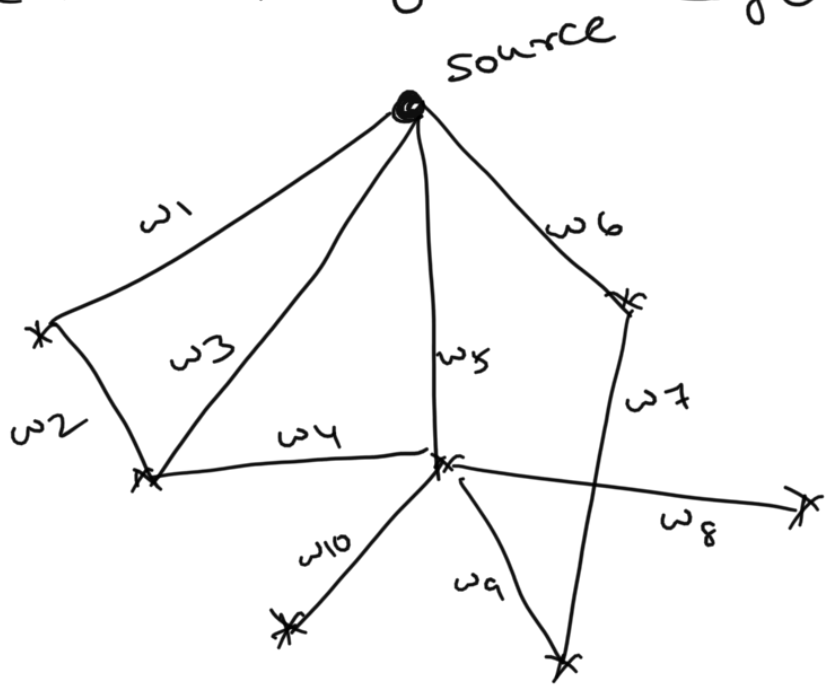
Dijkstra's algorithm

Input : weighted graph, source vertex

(no non negative edges)



Output : distances bet source and every other vertex

Notation : Graph G on n vertices

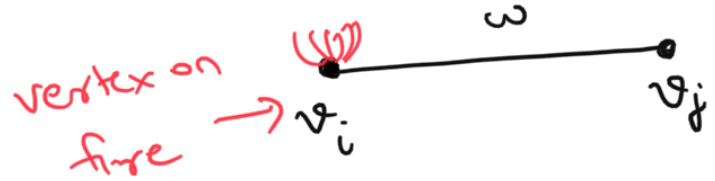
$V = \{1, 2, \ldots, n\}$ set of vertices

1 ↑ source vertex.

Idea :

Analogy

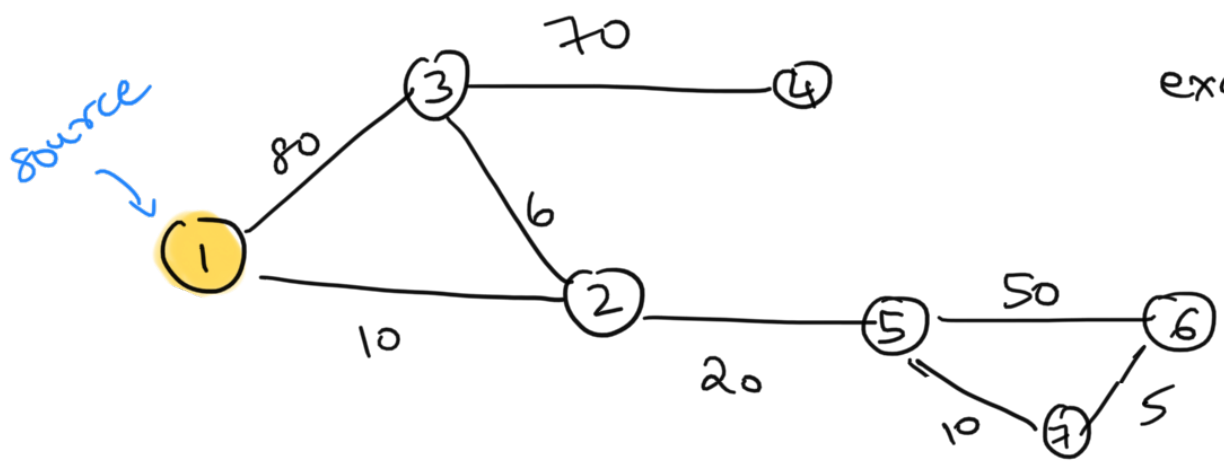
- vertices : oil depots
- edges : pipelines
- weight of edge : time taken for fire to reach v_j from v_i



- Set fire to source vertex
- fire will reach vertex closest to source first,

  second closet vertex to source next ....

Algorithm



example: G

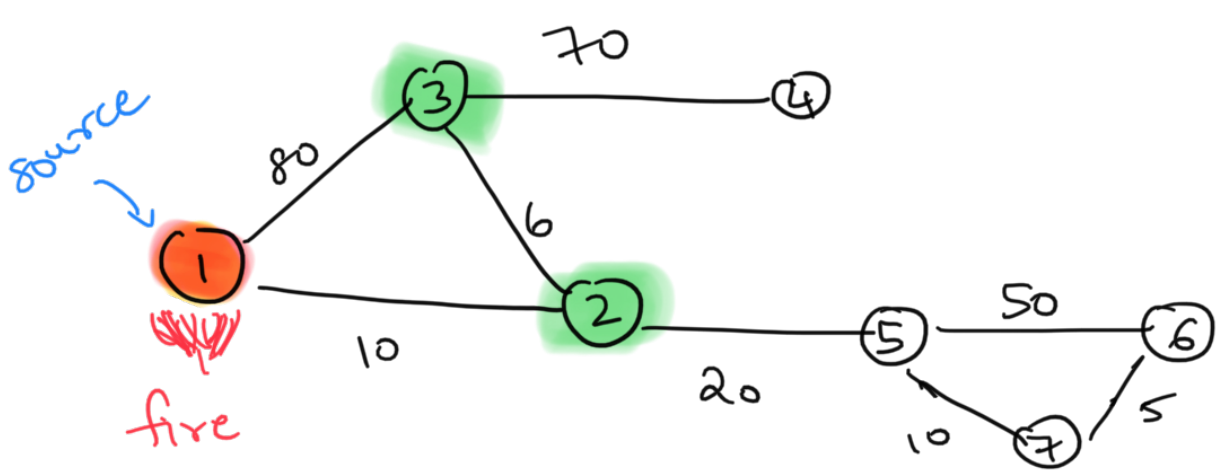
1) make 2 dictionaries (no fire yet)

Burned vertices (along with time of burning)

None

Potential time at which vertices will burn

vertex	potential time of burning
1	∞
2	∞
.	.
.	.
7	∞

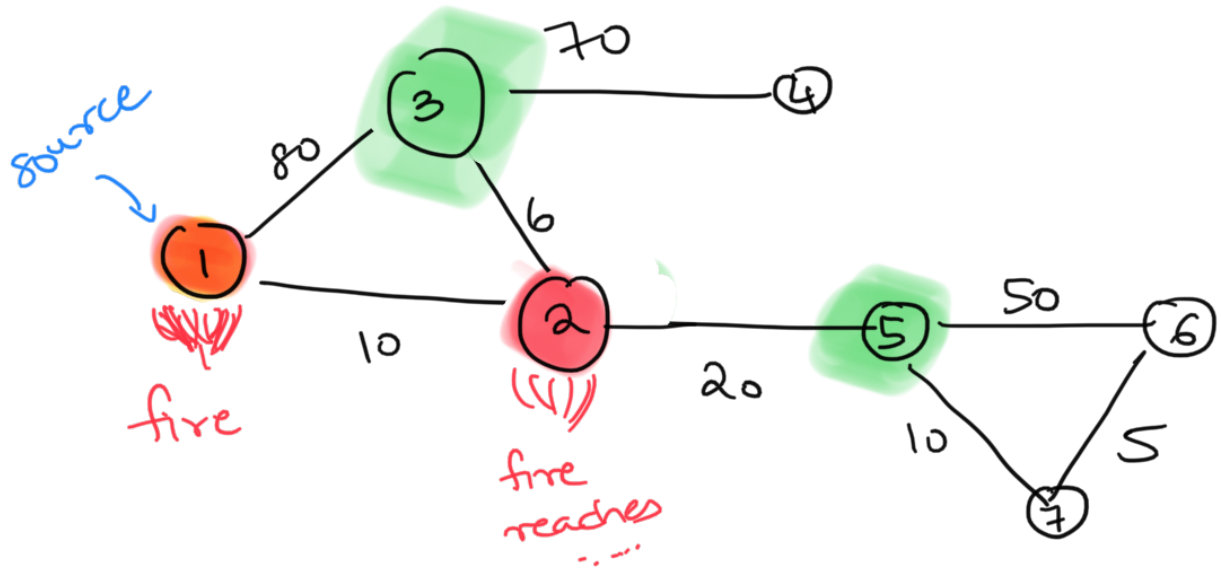


II) * Set fire to source vertex at time t=0

* Delete burnt vertex from potential list and move it to burnt list

* Update potential times ...

Burned	Potential times of burning
vertex 1 at t=0	vertex | time 2 | 10 3 | 80 4 | "∞" ...



III) * Pick vertex with shortest potential burning time t_1

* Set fire to this vertex at $t = t_1$

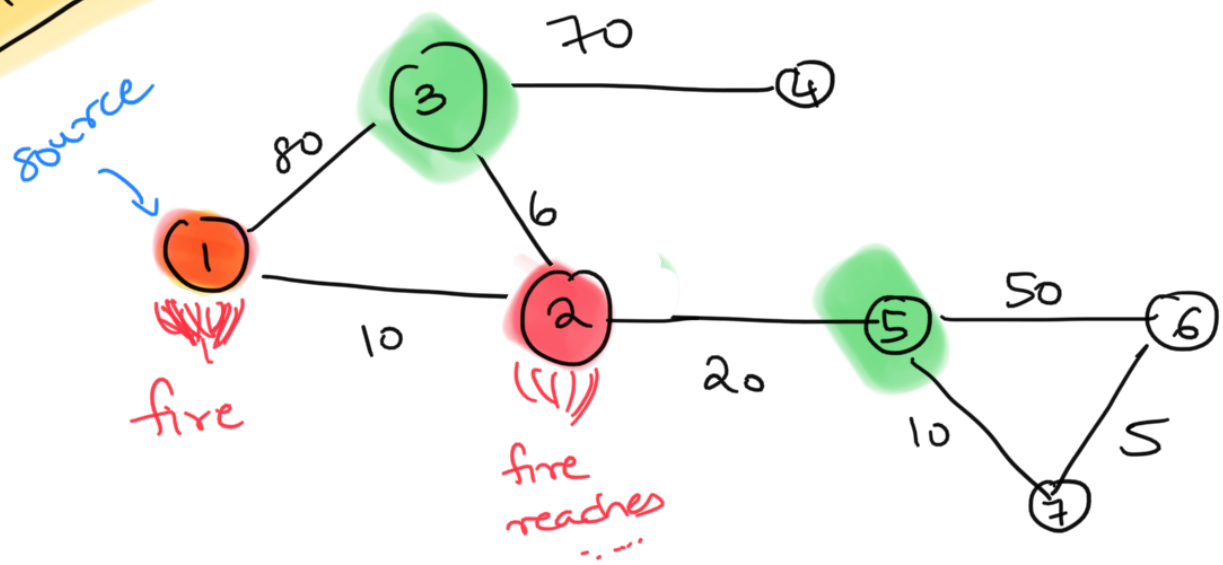
* Delete burnt vertex from potential list and move it to burnt list

★ Updak potential times ...

Burned	Potential times of burning
vertex 1 at t=0	
vertex 2 at t=10	

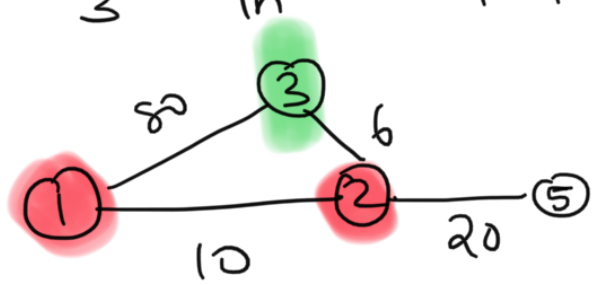
vertex	time
3	16
4	"∞"
5	30
6	"∞"
7	"∞"

explanation



→ Potential time for vertex 3 was 80 in step 2

→ can reach 3 in 10+6 = 16 time units via 2 ...

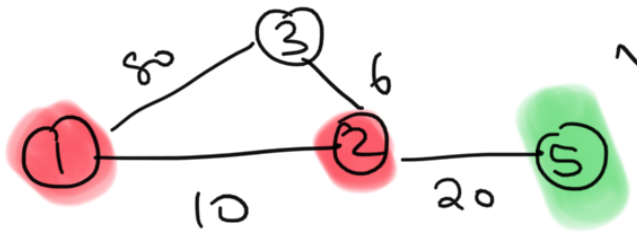


★ so update potential time for 3 as

$$\min(80, 16) = 16$$

→ Potential time for vertex 5 was ∞ in step 2

→ can reach 5 in 10 + 20 = 30 time units via 2...



★ so update potential for 5 as time

$$\min(\infty, 30) = 30$$

IV Repeat step III until either list of vertices with potential times is empty / min value is "∞".

↑ (means all other vertices are unreachable from source vertex)

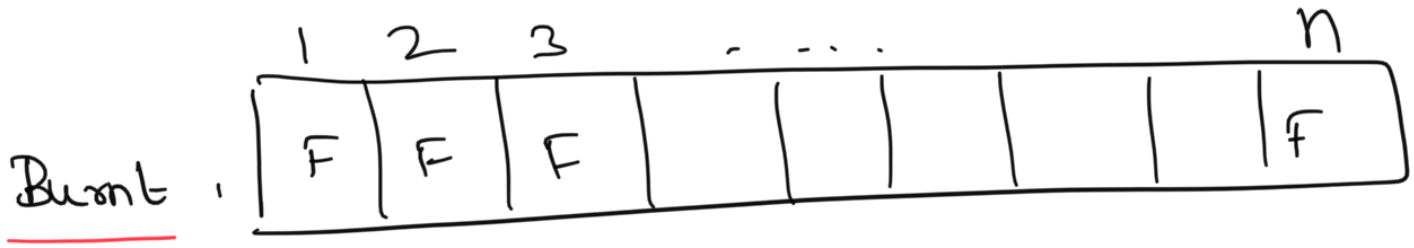
V shortest path from source to vertex v is Burnt[v]

↑ time at which 'v' was burnt

Comment about implementation:

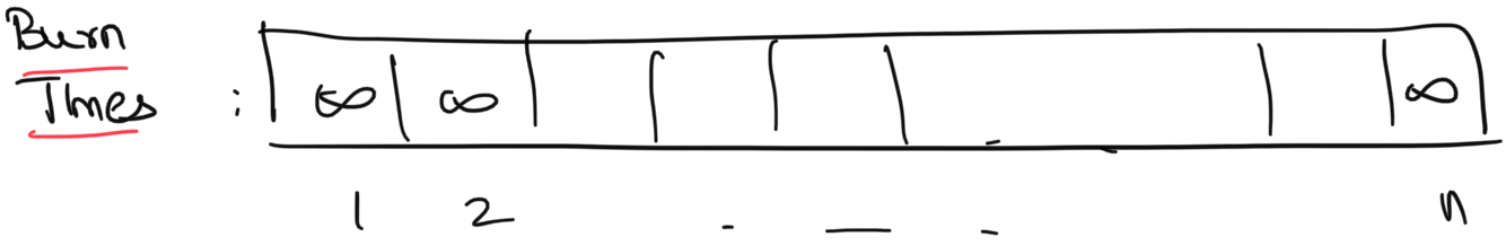
Will see more efficient implementation using heaps later …

→ Instead of deleting vertex with min potential time and adding it to burnt list .... can also do the foll



True / False list

if you burn a vertex, make its entry T



List of Burn times

→ while taking min, take min of burn times

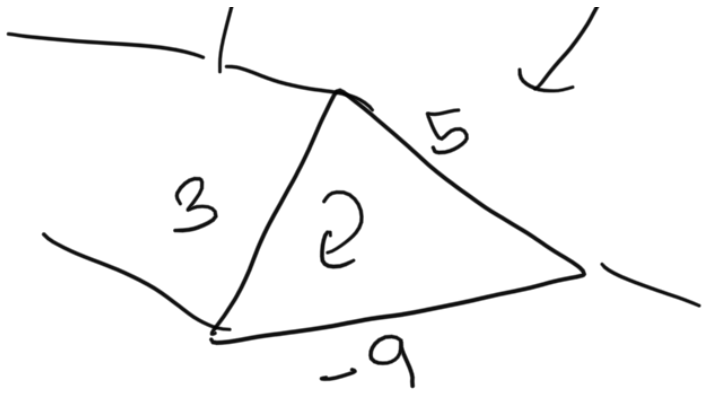
over unburnt vertices only.

→ while updating, update burn times

over unburnt vertices only

Negative weights

Doesn't work if negative edges present



if no negative cycle, but negative edges

??

→ Use Bellman-Ford/

→ Floyd-Warshall all pairs shortest path.

if negative cycle can loop again and again to set "$-\infty$" path length ⊗

Complexity analysis

→ you end up burning each vertex once (if graph connected)

(ie) there are $|V|$ iterations of step III

→ In order to burn a vertex

$O(|V|)$ { - you need to pick min potential burning time amongst $\leq |V|$ burning times

$O(d(v))$ ↑ degree of v { - update the burning times for neighbours of vertex to be burnt v

So overall

→ $|V|^2$ + $\sum_{v \in V} d(v)$ } operations

for picking min / all iterations →

for updates / all iterations ↓

$\sum_{v \in V} d(v) = 2|E|$

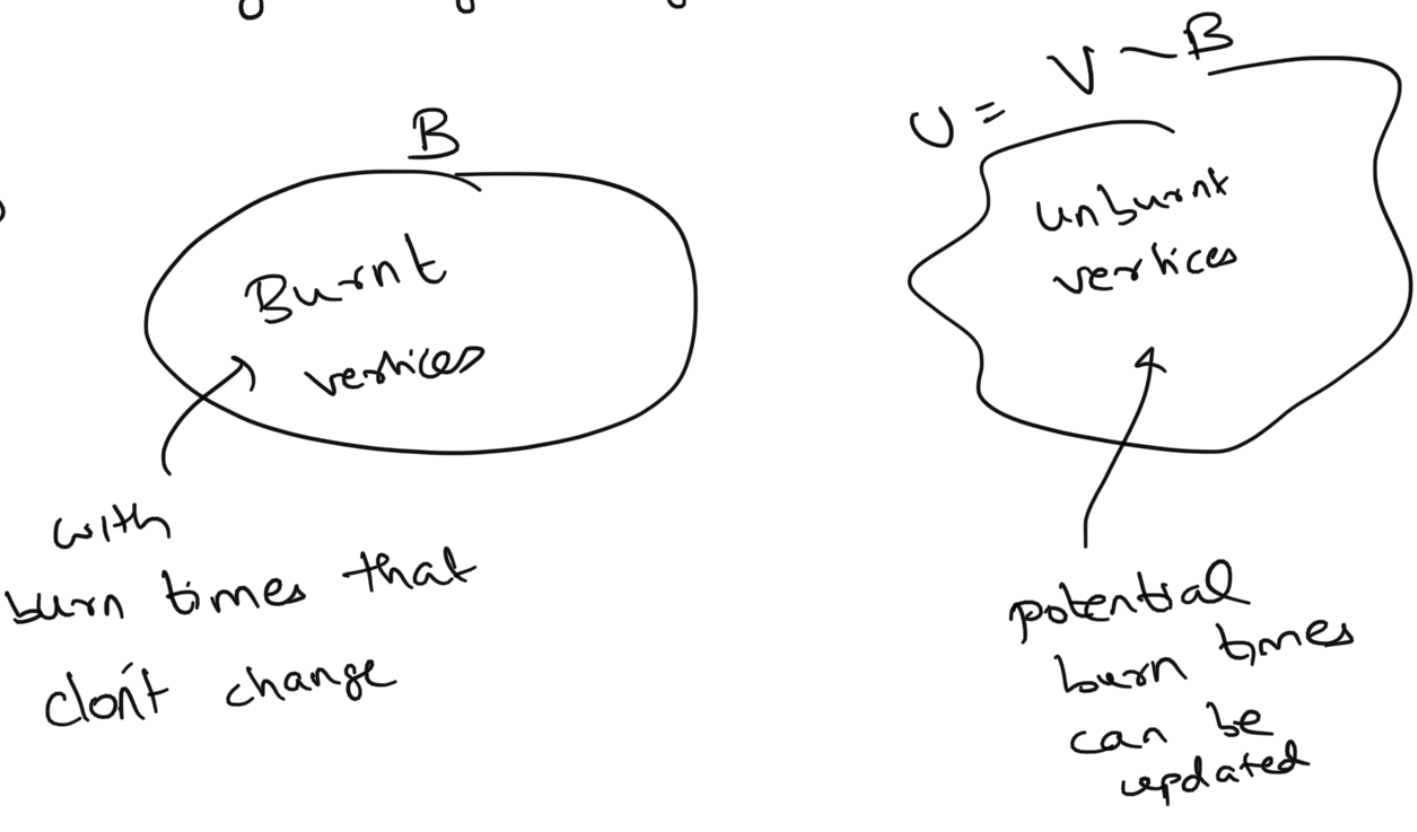
As $|E| \leq \binom{|V|}{2} = O(|V|^2)$,

overall complexity is $O(|V|^2)$

Why does Dijkstra work?

→ "greedy algorithm"

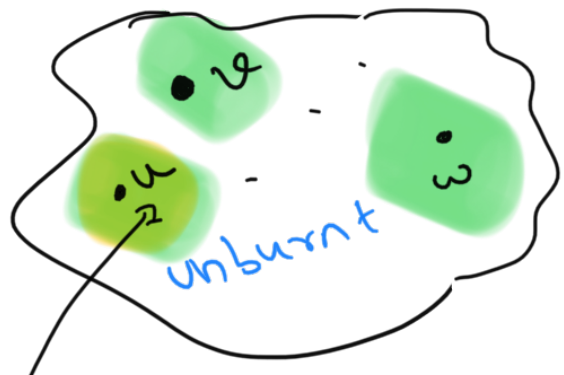
→



→ Assume by induction on $|B|$, that burn times for $v \in B$ give correct answer (record shortest paths)

→ now

Assume by induction Burnt[a] is correct $\forall a \in B$



$V \setminus B$

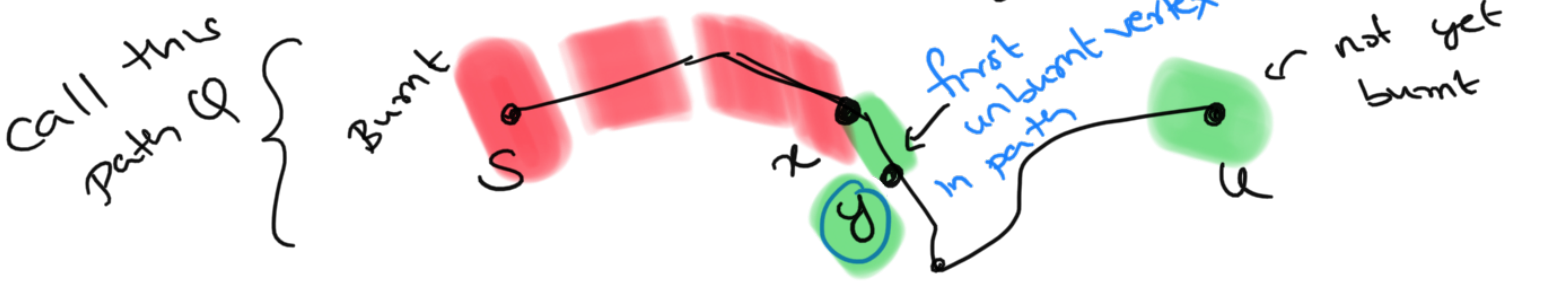
$d(v)$ = updated potential burn times (via u)

this is the vertex we are going to burn now

→ Let u = $\min_v d(v)$

Claim: shortest path from s to u $\geq d(u)$

Pf if not, say shortest path



has length $< d(u)$.

$$l(Q) \geq l(s \rightarrow x) + x - y \quad (*)$$

edge weight

$$= d(x) + x - y \qquad (*)$$

↑ By induction!

now y is adjacent to a burnt vertex x,

so $d(y)$ would have been updated already as $\min(d(y), d(x) + x-y \text{ edge})$

$\leq \ell(\varphi)$ therefore by $(*)$

But $u \leftarrow$ we are picking to be $\min_{v \in V \setminus B} d(v)$

so $d(u) \leq d(y) \leq \ell(\varphi)$

$< d(u)$

↑ our hypothesis

a contradiction...