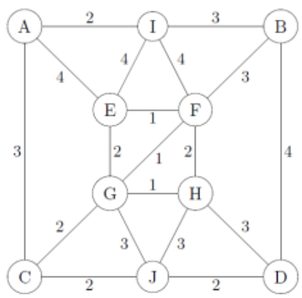
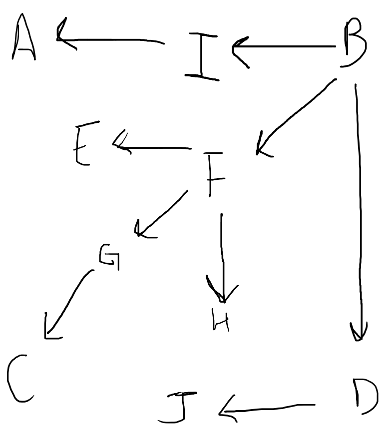
CSCB63 TUT4 WEEK7
andrewyk.leung@mail.utoronto.ca
https://snoopysnipe.github.io/ta/b63w21/

Perform Dijkstra's algorithm on the following graph - choose B as your start vertex:





v	d[v]	p[v]
~~A~~ 8	~~∞~~ 2 5	I
~~B~~ 1	0	NIL (null)
~~C~~ 9	~~∞~~ 5 6	G
~~D~~ 4	~~∞~~ 1 4	B
~~E~~ 6	~~∞~~ 2 ~~7~~ 3 4	~~I~~ 3 F
~~F~~ 3	~~∞~~ 1 3	B
~~G~~ 5	~~∞~~ 3 4	F
~~H~~ 7	~~∞~~ 3 5	F
~~I~~ 2	~~∞~~ 1 3	B
~~J~~ 10	~~∞~~ 4 6	D

Dijkstra's Correctness

Let T_s be distance tree constructed by Dijkstras starting at s

Let O_s be an optimal distance tree rooted at s

Let edges $e_1, \ldots, e_m$ be ordered according to how they are added to T_s

Consider the first edge $e_i = (u, v)$ such that

$e_i \in T_s$ and $e_i \notin O_s$ (ie: the first different edge between the two trees)

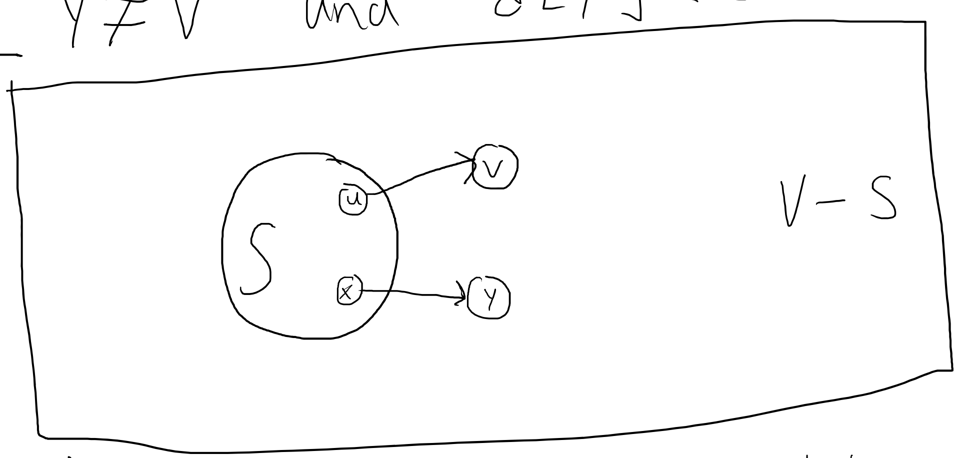
The $e_1, \dots, e_{i-1} \in T_s$ and let S be set of vertices added up to this point (ie: all endpoints of $e_1, \dots, e_{i-1}$)

Then each node in S has minimum path distance to s (the starting vertex)

Since $(u,v) \notin O_s$ there must exist a shorter path p from s to v

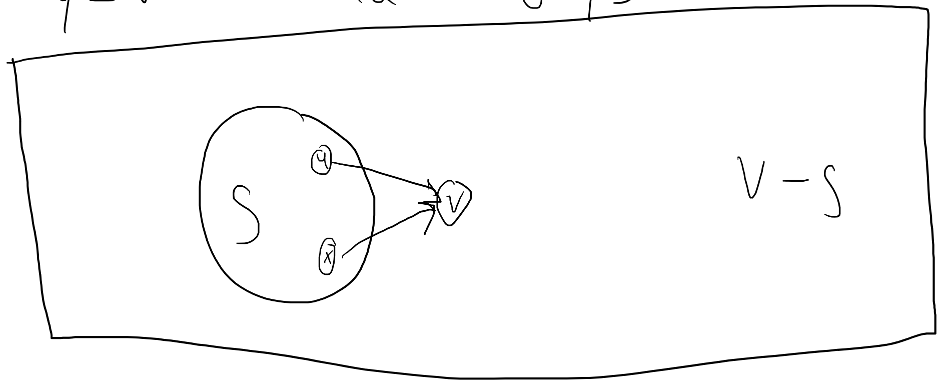
Consider edge $e_j = (x, y)$, $j > i$ on p that has one endpoint in S and one in $V-S$ $(x \in S, y \in V-S)$

Case 1: $y \neq v$ and $d[y] < d[v]$



then our algorithm would have chosen e_j next and not e_i

Case 2a: $y = v$ and $d_O[y] < d_T[v]$



then Dijkstra would also have chosen e_j instead of e_i

Case 2b: $y = v$ and $d_O[y] = d_T[v]$

then (x, v) can be swapped with (u, v) in O_s and now O_s looks closer to T_s

If you repeat this, eventually $T_s = O_s$

thus Dijkstras produces optimal result