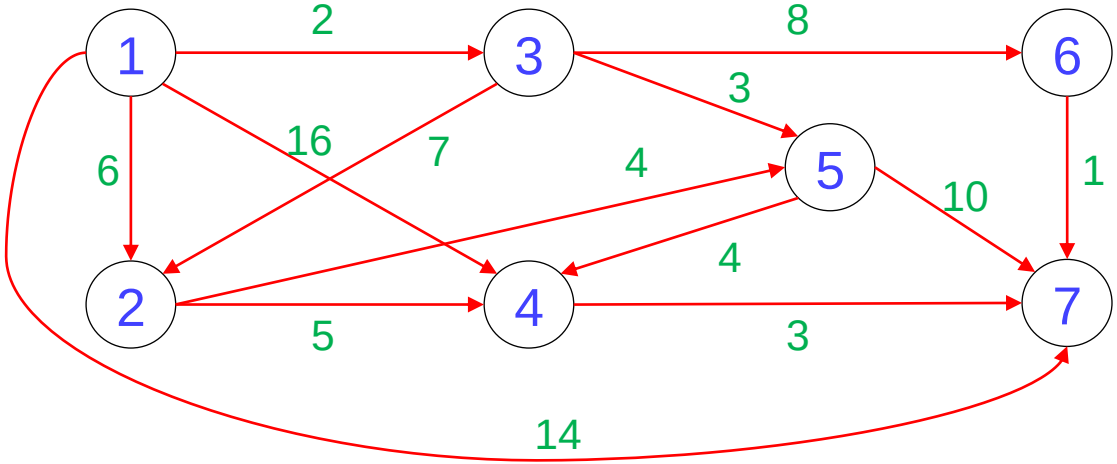
Dijkstra's Shortest Path Algorithm

Dr. Shrutilipi Bhattacharjee, Assistant Professor, Dept. of IT, NIT Karnataka, India

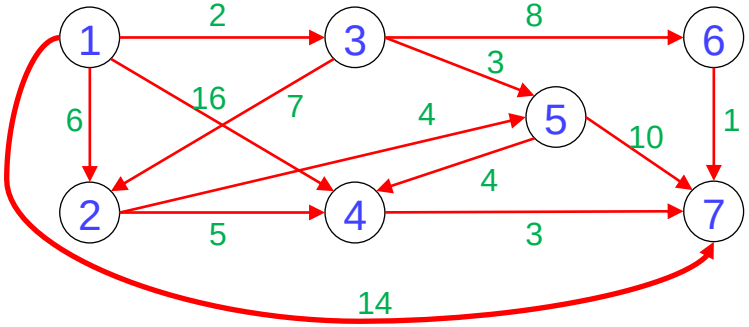
Shortest Path Problems

- Lets assume that the graph ***G*** is directed
- Directed weighted graph
- Each edge has a length/cost $L : E \rightarrow IR^{+}$
- The problem is to find the shortest path from ***s*** (source) to any ***t*** (destination)

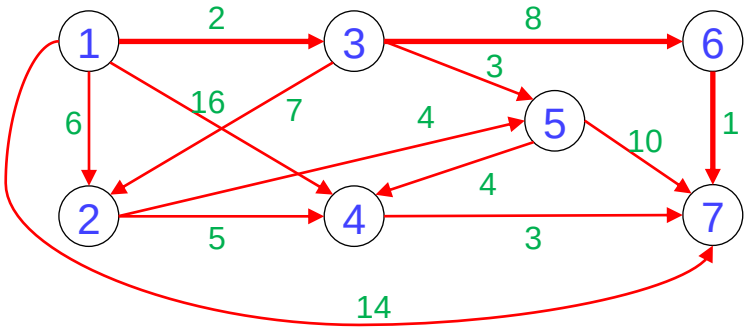


Shortest Path Problems

- Single source single destination
- Single source all destinations
- All pairs (every vertex is a source and destination)
- Assume edge costs (lengths) are ≥ 0
- Path length is sum of weights of edges on path
- The vertex at which the path begins is the *source* vertex
- The vertex at which the path ends is the *destination* vertex

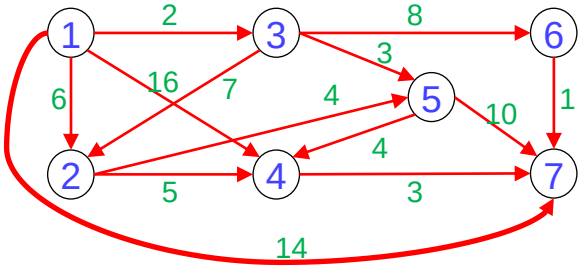


A path from ***1*** to ***7***: Path length is ***14***

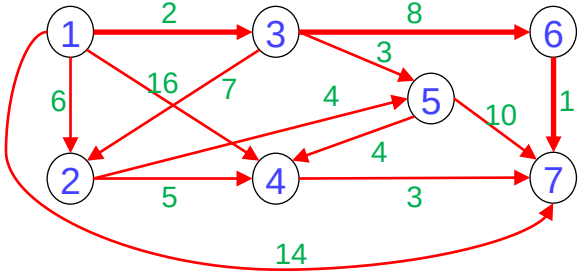


A path from ***1*** to ***7***: Path length is ***11***

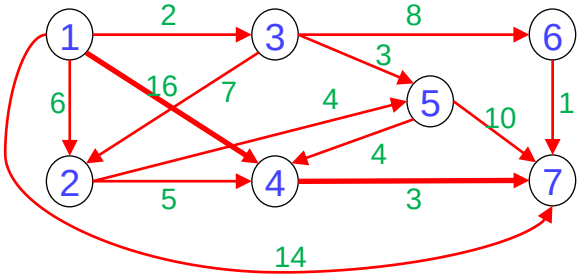
How Many Possible Paths



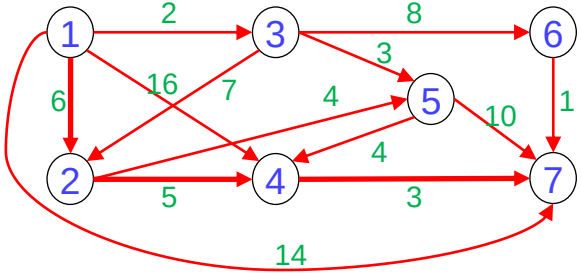
A path from ***1*** to ***7***: Path length is ***14***



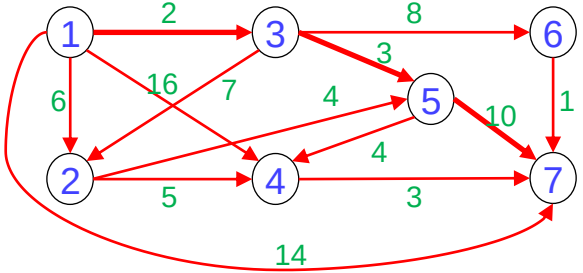
A path from ***1*** to ***7***: Path length is ***11***



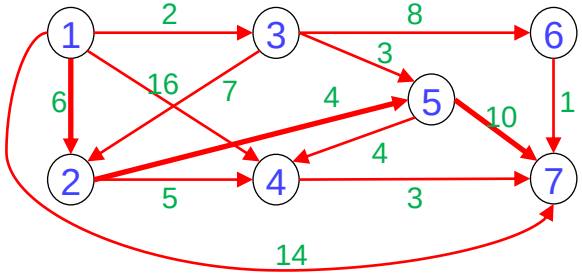
A path from ***1*** to ***7***: Path length is ***19***



A path from ***1*** to ***7***: Path length is ***14***



A path from ***1*** to ***7***: Path length is ***15***

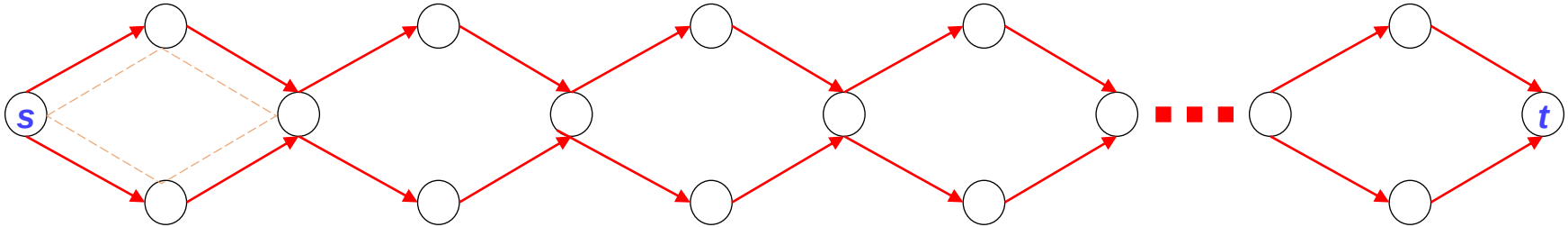


A path from ***1*** to ***7***: Path length is ***20***

Naïve Approach

Can we always list out all the paths in the graph and just take whichever is the shortest, compute the length of each part and then compute the shortest?

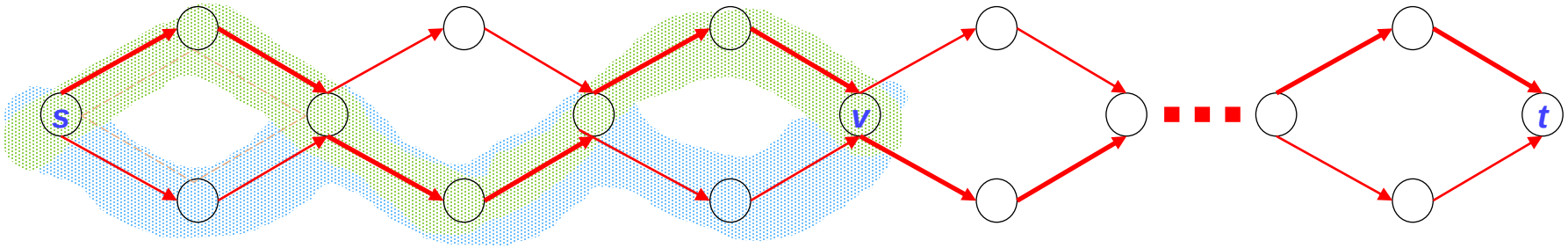
- Lets take a very large graph
- Continue until destination is reached



- If there are ***k*** number of diamonds, how many vertices are? ***3k + 1***
- How many paths are there between ***s*** and ***t***? 2^k
- What is the value of ***k***? ≈ ***V/3***
- Then the number of paths between ***s*** and ***t***? $2^{V/3}$

Property of Shortest Paths

- Suppose this (red bold edges) is the shortest path from ***s*** to ***t***
- Let ***v*** be any arbitrary vertex in this path
- Then this (green shaded path) is also a shortest path from ***s*** to ***v***
  - Lets try to prove it by contradiction
  - Suppose there is a shorter path from ***s*** to ***v*** (blue shaded path)
  - Then, cost[blue(***s, v***)] + cost[(***v, t***)] ≤ cost[green(***s, v***)] + cost[(***v, t***)]
  - Then, this (red bold edges) is **NOT** the shortest path from ***s*** to ***t***
  - **Contradiction !!!**



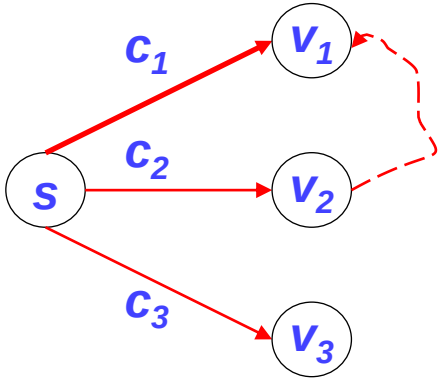
Property of Shortest Paths

- We will compute the shortest path from ***s*** to every vertex in ***G***
- This problem is also called as **single source shortest path** (SSSP) problem
- Now, which vertex or the shortest path from ***s*** to which vertex is immediate?
  - Minimum of the adjacent ones
  - So, the shortest path from ***s*** to v_1 is c_1
  - Otherwise, cost[(s, v_1)] = cost[(s, v_2)] + cost[(v_2, v_1)]

    $$c_1 = c_2 + \delta$$

    $$c_1 \geq c_2 \quad [\text{as } \delta \geq 0]$$

  - **Contradiction !!!**

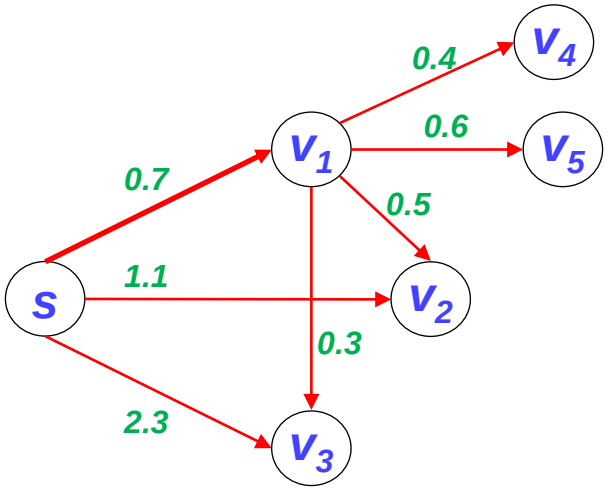


$c_1 < c_2 < c_3$

- Therefore, distance label ***d*** for v_1, $d[v_1] = c_1$
- Similarly, $d[v_2] \leq c_2$ and $d[v_3] \leq c_3$

Property of Shortest Paths

- What to do next?
- $d[v_1] = c_1 = 0.7$
- Similarly, $d[v_2] \leq 1.1$ and $d[v_3] \leq 2.3$
- After getting the path (s, v_1), we have found a better path: $d[v_3] \leq 1$
- But, have we got any better path for v_2?
- Similarly, $d[v_4] \leq 1.1$ and $d[v_3] \leq 1.3$
- Now, let me take the smallest cost ($d[v_3] = 1$) and this is correct
- Till now, this is a claim, and we have to prove its correctness



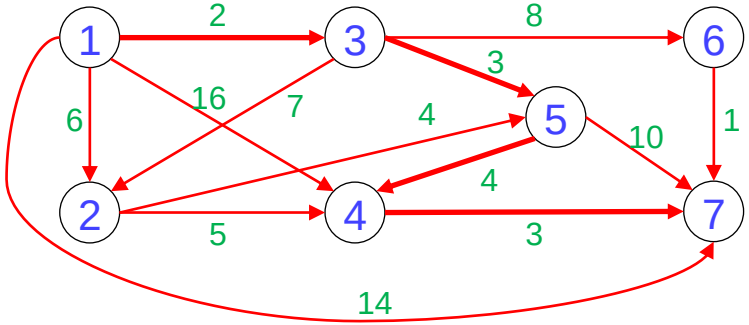
Single-Source Single Destination

Possible greedy algorithm

- Leave source vertex using cheapest/shortest edge
- Leave new vertex using cheapest edge subject to the constraint that a new vertex is reached
- Continue until destination is reached

Single Source All Destinations

- Need to generate up to ***N*** (***N*** is number of vertices) paths (including path from source to itself)
- Greedy method:
  - Construct these up to ***N*** paths in order of increasing length
  - Assume edge costs (lengths) are ≥ 0
  - So, no path has length < 0
  - First shortest path is from the source vertex to itself
  - The length of this path is 0

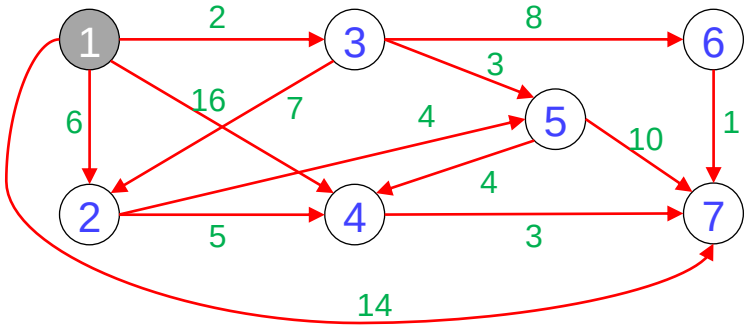


Path length is ***12***

Not shortest path. Algorithm doesn't work!

Greedy Single Source All Destinations

Path	Length
1	0
1 → 3	2
1 → 3 → 5	5
1 → 2	6
1 → 3 → 5 → 4	9
1 → 3 → 6	10
1 → 3 → 6 → 7	11



- Each path (other than first) is a one edge extension of a previously explored shortest path
- Next shortest path is the shortest one edge extension of an already generated shortest path

Next Lecture

Dijkstra's Shortest Path Algorithm

Thank you for your attention...

Any question?

Contact:
Department of Information Technology, NITK Surathkal, India
6th Floor, Room: 13
Phone: +91-9477678768
E-mail: shrutilipi@nitk.edu.in, shrutilipi.bhattacharjee@tum.de