Approximate Solution

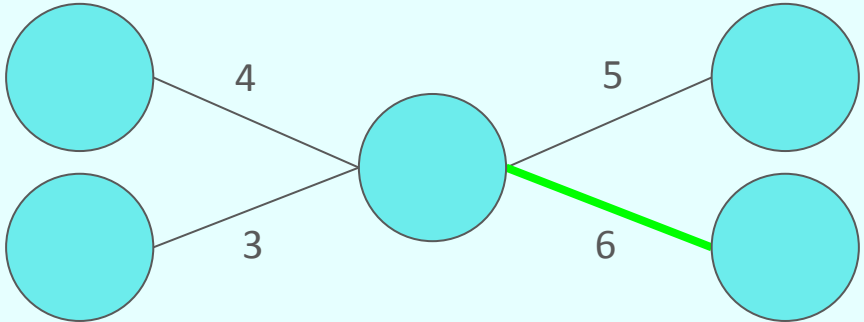
Algorithm (Approximate Greedy Longest Path):

For each vertex in the graph: → runs **n** times
  start a new path at that vertex
  mark it as visited

  While the current vertex has any unvisited neighbors: → ≤ **n** iterations (each vertex visited once)
    choose the unvisited neighbor connected by the heaviest edge → ≤ **n** (scans all neighbors)
    add that neighbor to the path
    mark it visited
    move to that neighbor

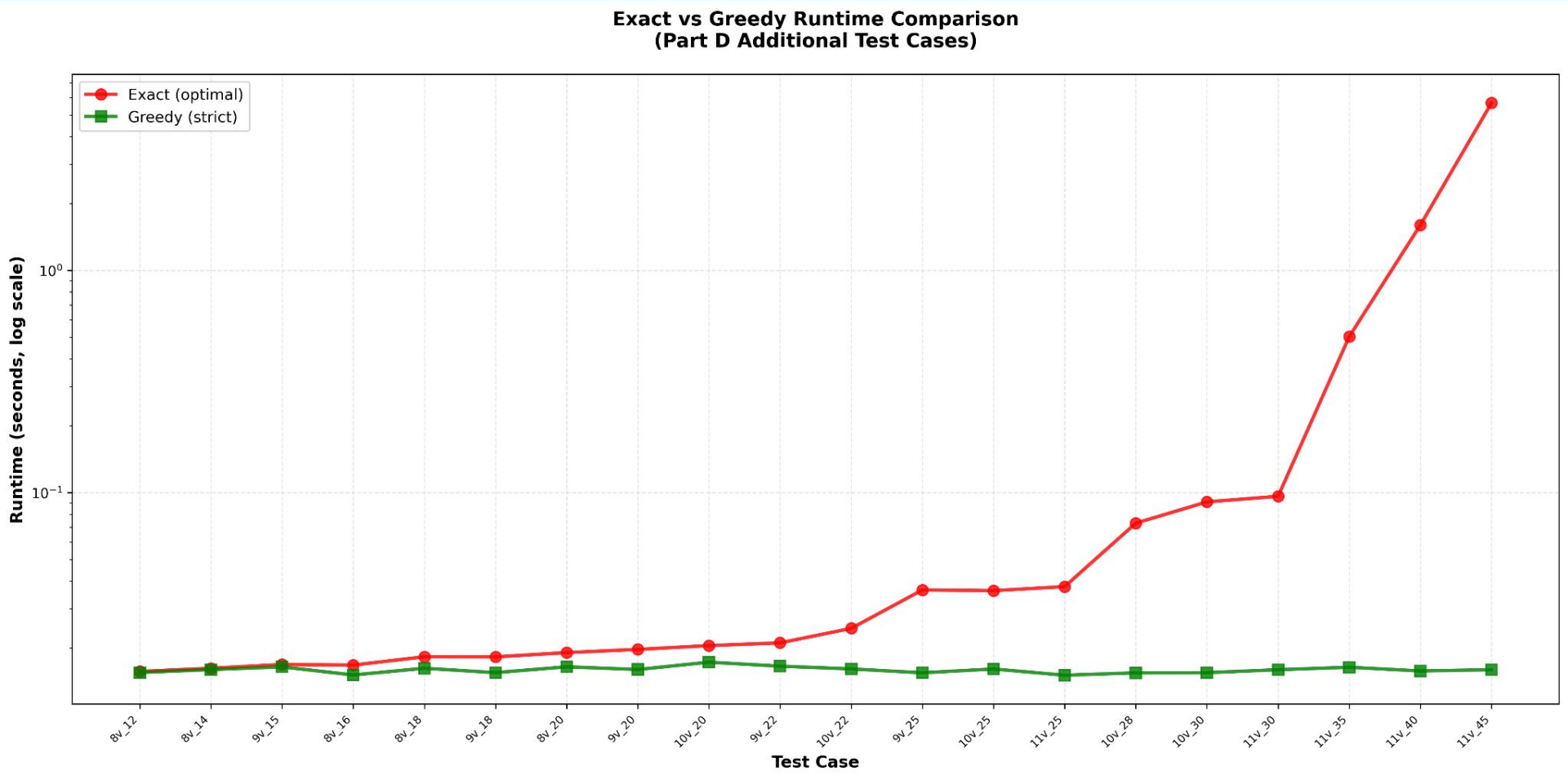
  keep track of the longest path found across all starting vertices

return the longest path discovered

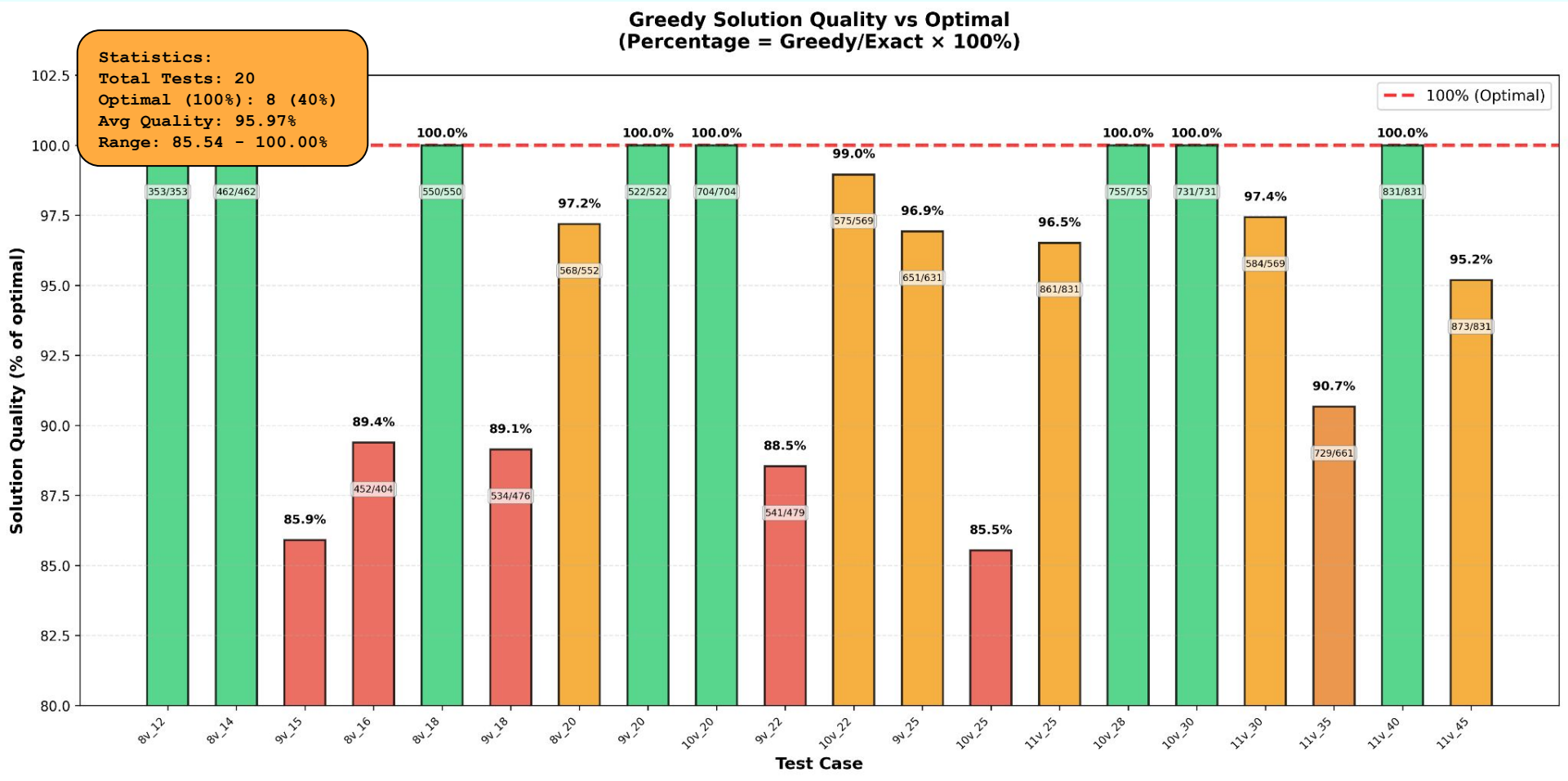


Runtime Analysis:

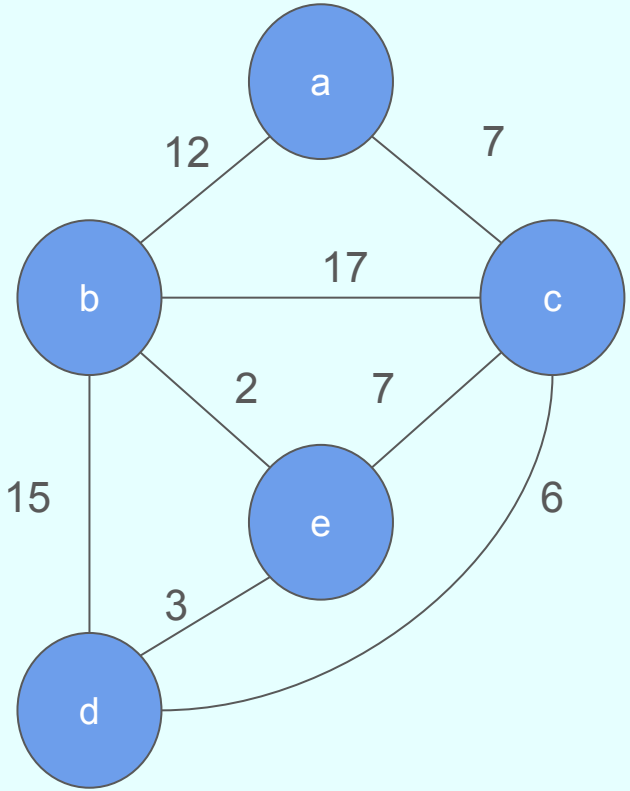
$O(n^3)$



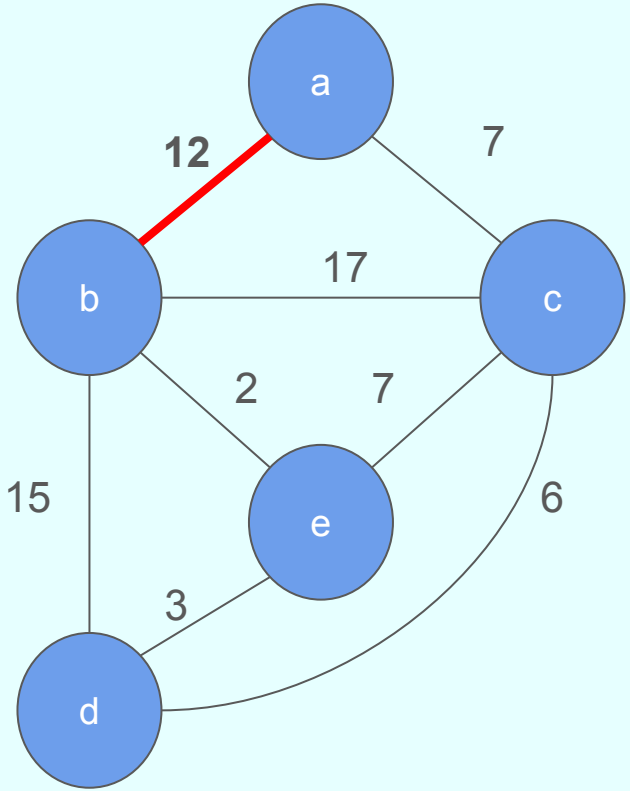
8v_16 → 8 vertices, 16 edges


Greedy Solution Quality vs Optimal
(Percentage = Greedy/Exact × 100%)
Statistics:
Total Tests: 20
Optimal (100%): 8 (40%)
Avg Quality: 95.97%
Range: 85.54 - 100.00%
100% (Optimal)
Solution Quality (% of optimal)
Test Case
8v_12
8v_14
9v_15
8v_16
8v_18
9v_18
8v_20
9v_20
10v_20
9v_22
10v_22
9v_25
10v_25
11v_25
10v_28
10v_30
11v_30
11v_35
11v_40
11v_45
100.0%
100.0%
100.0%
85.9%
89.4%
100.0%
89.1%
97.2%
100.0%
100.0%
88.5%
99.0%
96.9%
85.5%
96.5%
100.0%
100.0%
97.4%
90.7%
100.0%
95.2%
353/353
462/462
452/404
550/550
534/476
568/552
522/522
704/704
541/479
575/569
651/631
861/831
755/755
731/731
584/569
729/661
831/831
873/831

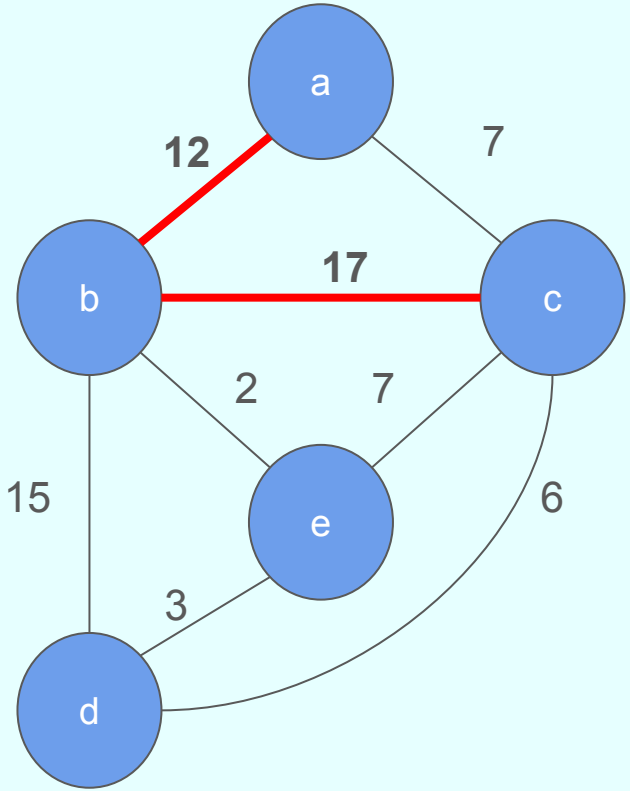
Why Greedy Can Fail



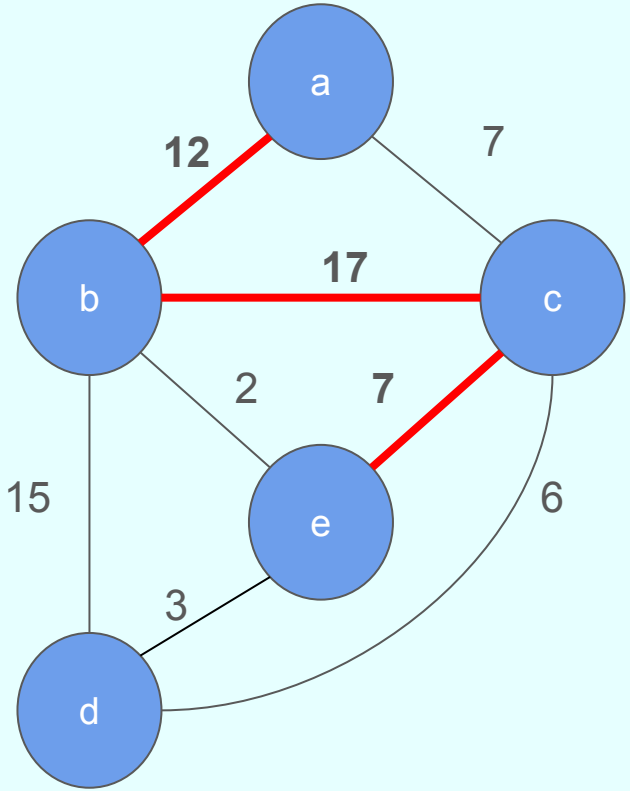
Greedy starting at a



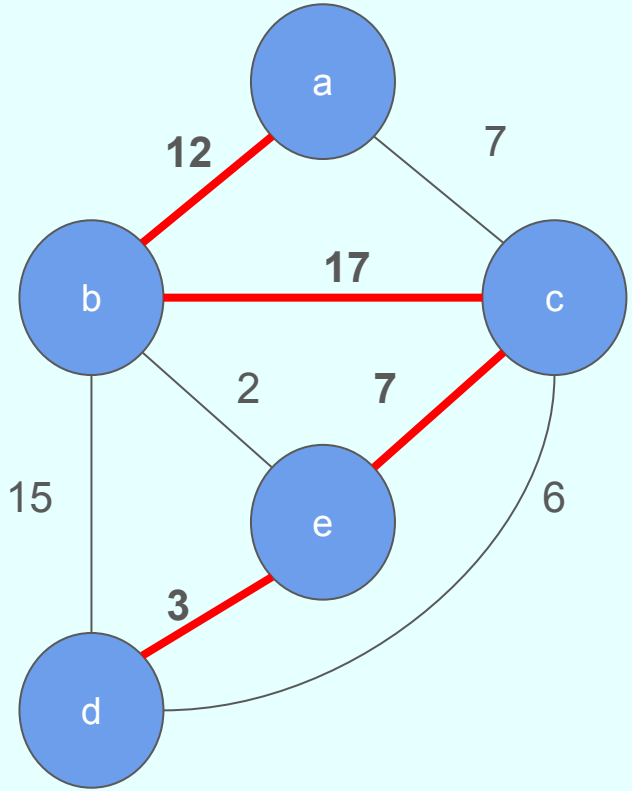
Greedy starting at a



Greedy starting at a

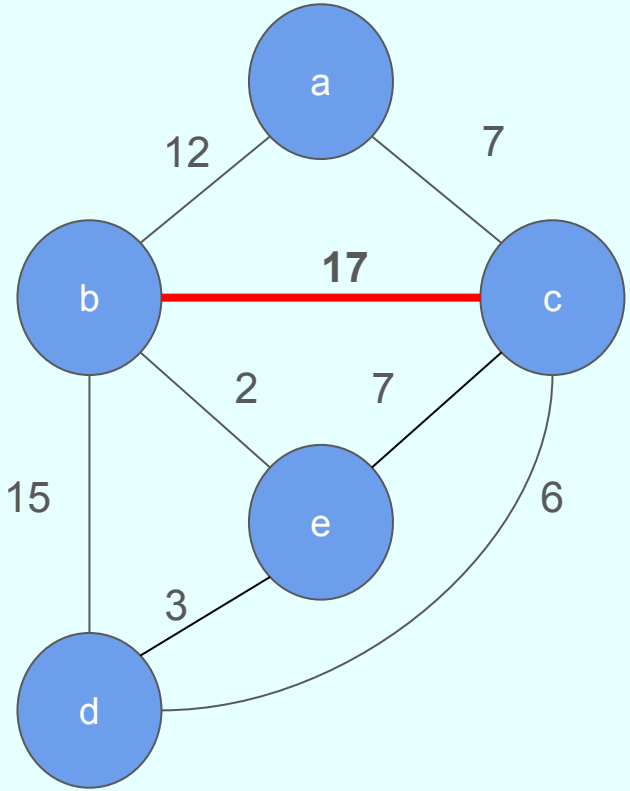


Greedy starting at a

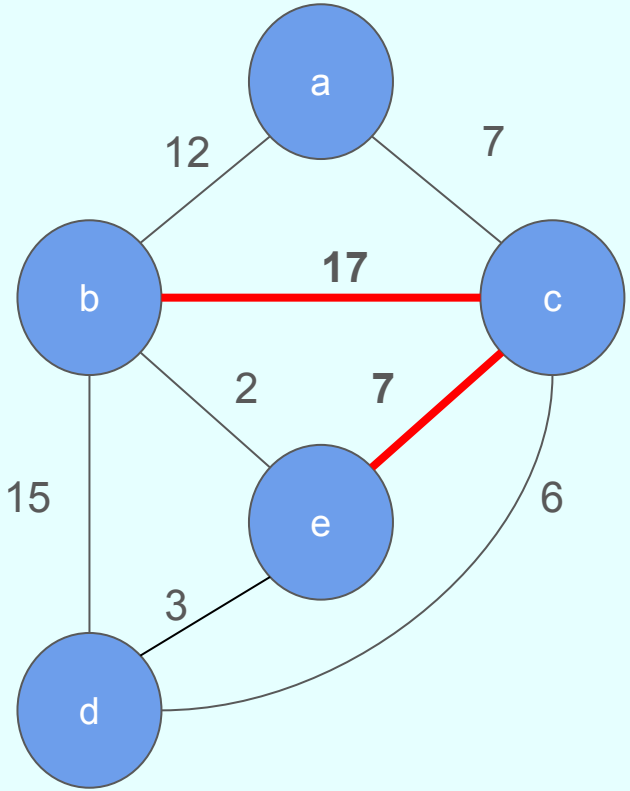


39 total

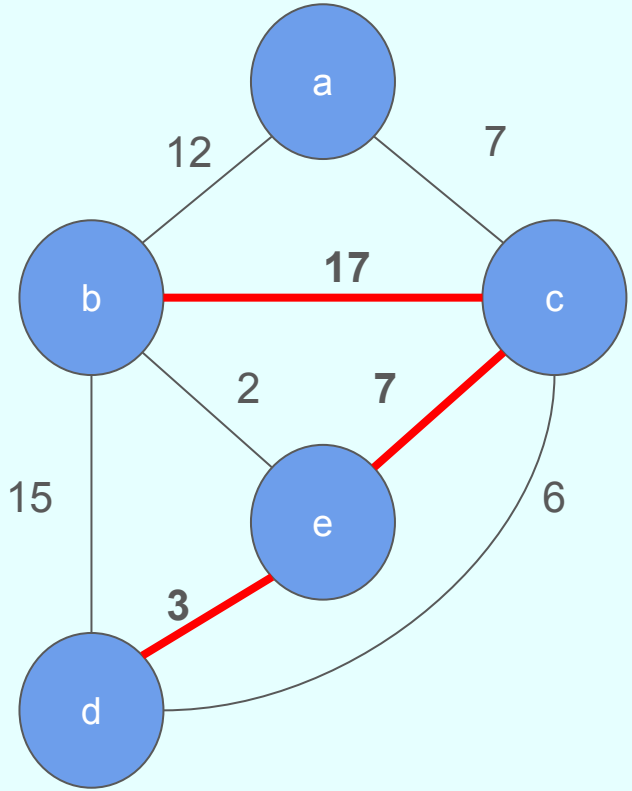
Greedy starting at b



Greedy starting at b

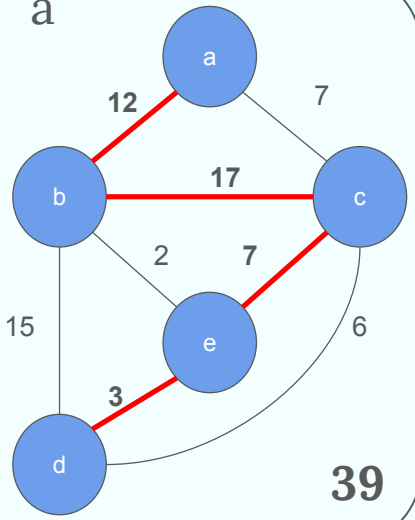


Greedy starting at b



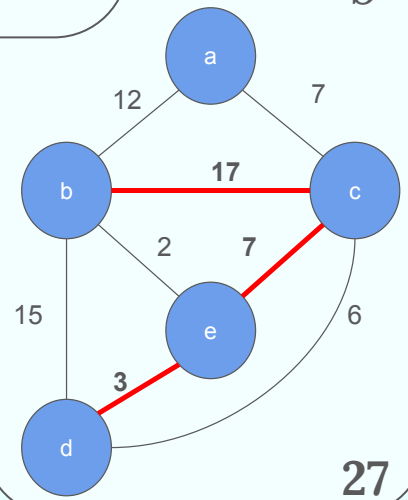
27 total

a



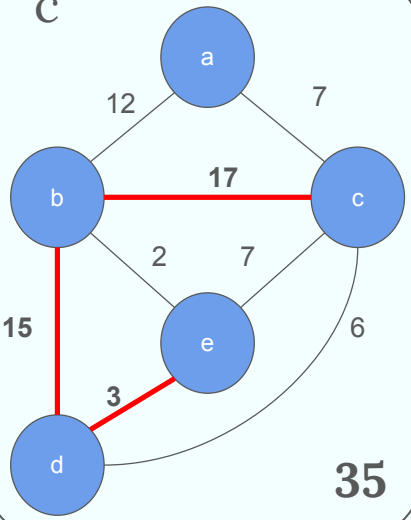
39

b



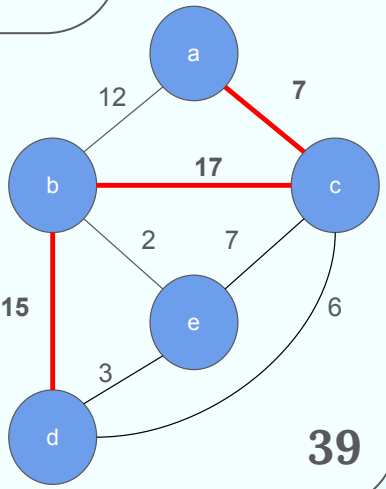
27

c



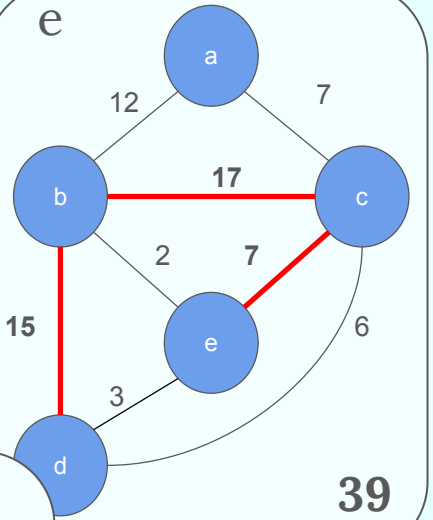
35

d



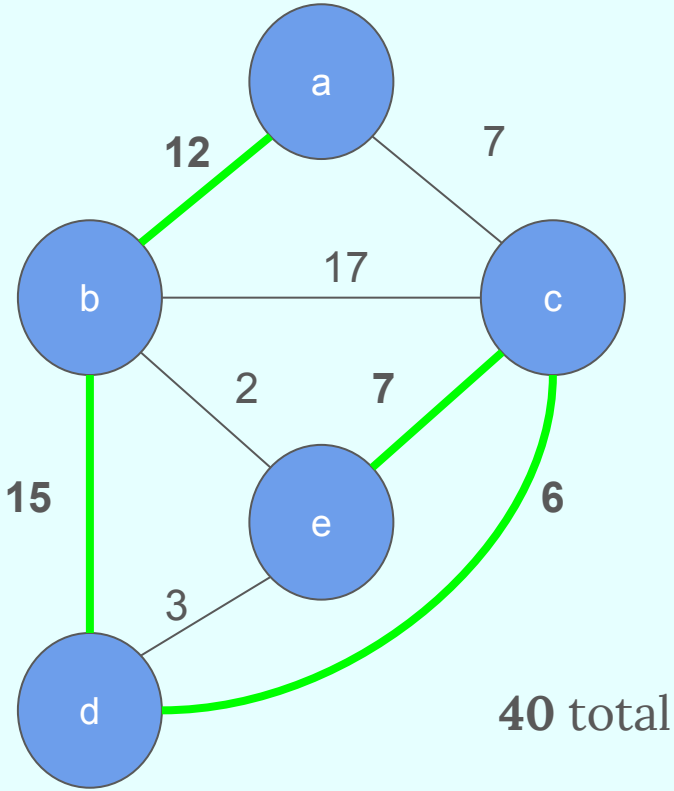
39

e



39

Optimal solution



40 total