----------

GRAPH 1:

----------

[1] graph traversal costs and paths,

brute force connected edge tracing:

0.015255 seconds (2.59 k allocations: 141.899 KiB)

2×2 DataFrames.DataFrame

│ Row │ cost │ path │

├─────┼──────┼───────────┤

│ 1 │ 8 │ [1, 2, 4] │

│ 2 │ 8 │ [1, 3, 4] │

[2] graph minimum cost and optimal path,

Dijkstra's algorithm:

(8, [1, 2, 4])

0.000145 seconds (105 allocations: 7.859 KiB)

----------

GRAPH 2:

----------

[1] graph traversal costs and paths,

brute force connected edge tracing:

0.019887 seconds (2.87 k allocations: 163.745 KiB)

2×2 DataFrames.DataFrame

│ Row │ cost │ path │

├─────┼──────┼──────────────┤

│ 1 │ 19 │ [1, 2, 4] │

│ 2 │ 17 │ [1, 2, 3, 4] │

[2] graph minimum cost and optimal path,

Dijkstra's algorithm:

(17, [1, 2, 3, 4])

0.000155 seconds (111 allocations: 8.016 KiB)

----------

GRAPH 3:

----------

[1] graph traversal costs and paths,

brute force connected edge tracing:

0.000031 seconds (62 allocations: 3.734 KiB)

3×2 DataFrames.DataFrame

│ Row │ cost │ path │

├─────┼──────┼──────────────┤

│ 1 │ 18 │ [1, 2, 3, 5] │

│ 2 │ 17 │ [1, 2, 4, 5] │

│ 3 │ 16 │ [1, 2, 5] │

[2] graph minimum cost and optimal path,

Dijkstra's algorithm:

(16, [1, 2, 5])

0.000098 seconds (113 allocations: 8.641 KiB)

----------

GRAPH 4:

----------

[1] graph traversal costs and paths,

brute force connected edge tracing:

0.006065 seconds (1.13 k allocations: 66.318 KiB)

4×2 DataFrames.DataFrame

│ Row │ cost │ path │

├─────┼──────┼────────────────────┤

│ 1 │ 61 │ [1, 2, 6] │

│ 2 │ 58 │ [1, 2, 3, 4, 5, 6] │

│ 3 │ 51 │ [1, 2, 5, 6] │

│ 4 │ 49 │ [1, 2, 4, 5, 6] │

[2] graph minimum cost and optimal path,

Dijkstra's algorithm:

(49, [1, 2, 4, 5, 6])

0.000117 seconds (136 allocations: 10.578 KiB)

----------

GRAPH 5:

----------

[1] graph traversal costs and paths,

brute force connected edge tracing:

0.000054 seconds (397 allocations: 18.188 KiB)

9×2 DataFrames.DataFrame

│ Row │ cost │ path │

├─────┼──────┼─────────────────┤

│ 1 │ 60 │ [1, 3, 6, 7] │

│ 2 │ 56 │ [1, 3, 2, 7] │

│ 3 │ 55 │ [1, 3, 2, 6, 7] │

│ 4 │ 52 │ [1, 2, 7] │

│ 5 │ 52 │ [1, 4, 5, 7] │

│ 6 │ 51 │ [1, 2, 6, 7] │

│ 7 │ 51 │ [1, 3, 5, 7] │

│ 8 │ 51 │ [1, 4, 5, 6, 7] │

│ 9 │ 50 │ [1, 3, 5, 6, 7] │

[2] graph minimum cost and optimal path,

Dijkstra's algorithm:

(50, [1, 3, 5, 6, 7])

0.000129 seconds (148 allocations: 13.125 KiB)

----------

GRAPH 6:

----------

[1] graph traversal costs and paths,

brute force connected edge tracing:

0.000099 seconds (1.25 k allocations: 47.906 KiB)

12×2 DataFrames.DataFrame

│ Row │ cost │ path │

├─────┼──────┼────────────────────┤

│ 1 │ 41 │ [1, 4, 9, 12, 14] │

│ 2 │ 36 │ [1, 5, 10, 13, 14] │

│ 3 │ 34 │ [1, 4, 9, 13, 14] │

│ 4 │ 34 │ [1, 5, 9, 12, 14] │

│ 5 │ 30 │ [1, 4, 8, 11, 14] │

│ 6 │ 27 │ [1, 3, 8, 11, 14] │

│ 7 │ 27 │ [1, 5, 9, 13, 14] │

│ 8 │ 27 │ [1, 6, 10, 13, 14] │

│ 9 │ 22 │ [1, 3, 7, 11, 14] │

│ 10 │ 21 │ [1, 4, 8, 12, 14] │

│ 11 │ 20 │ [1, 2, 7, 11, 14] │

│ 12 │ 18 │ [1, 3, 8, 12, 14] │

[2] graph minimum cost and optimal path,

Dijkstra's algorithm:

(18, [1, 3, 8, 12, 14])

0.000156 seconds (226 allocations: 25.422 KiB)

----------

GRAPH 7:

----------

[1] graph traversal costs and paths,

brute force connected edge tracing:

0.006879 seconds (11.53 k allocations: 433.990 KiB)

44×2 DataFrames.DataFrame

│ Row │ cost │ path │

├─────┼──────┼──────────────────────────────┤

│ 1 │ 29 │ [1, 2, 6, 7, 10, 19, 20] │

│ 2 │ 27 │ [1, 2, 6, 7, 10, 16, 20] │

│ 3 │ 27 │ [1, 2, 6, 9, 13, 16, 20] │

│ 4 │ 27 │ [1, 2, 6, 12, 15, 17, 20] │

│ 5 │ 27 │ [1, 3, 6, 7, 10, 19, 20] │

│ 6 │ 27 │ [1, 2, 6, 7, 10, 16, 19, 20] │

│ 7 │ 27 │ [1, 2, 6, 9, 13, 16, 19, 20] │

│ 8 │ 26 │ [1, 2, 6, 12, 15, 20] │

│ 9 │ 26 │ [1, 2, 6, 8, 11, 17, 20] │

│ 10 │ 26 │ [1, 2, 6, 8, 12, 15, 17, 20] │

│ 11 │ 25 │ [1, 2, 6, 8, 12, 15, 20] │

│ 12 │ 25 │ [1, 3, 6, 7, 10, 16, 20] │

│ 13 │ 25 │ [1, 3, 6, 9, 13, 16, 20] │

│ 14 │ 25 │ [1, 3, 6, 12, 15, 17, 20] │

│ 15 │ 25 │ [1, 3, 6, 7, 10, 16, 19, 20] │

│ 16 │ 25 │ [1, 3, 6, 9, 13, 16, 19, 20] │

│ 17 │ 24 │ [1, 3, 6, 12, 15, 20] │

│ 18 │ 24 │ [1, 3, 6, 8, 11, 17, 20] │

│ 19 │ 24 │ [1, 3, 6, 8, 12, 15, 17, 20] │

│ 20 │ 23 │ [1, 2, 6, 12, 15, 18, 20] │

│ 21 │ 23 │ [1, 3, 6, 8, 12, 15, 20] │

│ 22 │ 22 │ [1, 2, 6, 8, 12, 15, 18, 20] │

│ 23 │ 21 │ [1, 7, 10, 19, 20] │

│ 24 │ 21 │ [1, 2, 5, 11, 17, 20] │

│ 25 │ 21 │ [1, 2, 5, 8, 11, 17, 20] │

│ 26 │ 21 │ [1, 3, 6, 12, 15, 18, 20] │

│ 27 │ 21 │ [1, 4, 6, 7, 10, 19, 20] │

│ 28 │ 21 │ [1, 2, 5, 8, 12, 15, 17, 20] │

│ 29 │ 20 │ [1, 2, 5, 8, 12, 15, 20] │

│ 30 │ 20 │ [1, 3, 6, 8, 12, 15, 18, 20] │

│ 31 │ 19 │ [1, 7, 10, 16, 20] │

│ 32 │ 19 │ [1, 7, 10, 16, 19, 20] │

│ 33 │ 19 │ [1, 4, 6, 7, 10, 16, 20] │

│ 34 │ 19 │ [1, 4, 6, 9, 13, 16, 20] │

│ 35 │ 19 │ [1, 4, 6, 12, 15, 17, 20] │

│ 36 │ 19 │ [1, 4, 6, 7, 10, 16, 19, 20] │

│ 37 │ 19 │ [1, 4, 6, 9, 13, 16, 19, 20] │

│ 38 │ 18 │ [1, 4, 6, 12, 15, 20] │

│ 39 │ 18 │ [1, 4, 6, 8, 11, 17, 20] │

│ 40 │ 18 │ [1, 4, 6, 8, 12, 15, 17, 20] │

│ 41 │ 17 │ [1, 4, 6, 8, 12, 15, 20] │

│ 42 │ 17 │ [1, 2, 5, 8, 12, 15, 18, 20] │

│ 43 │ 15 │ [1, 4, 6, 12, 15, 18, 20] │

│ 44 │ 14 │ [1, 4, 6, 8, 12, 15, 18, 20] │

[2] graph minimum cost and optimal path,

Dijkstra's algorithm:

(14, [1, 4, 6, 8, 12, 15, 18, 20])

0.000292 seconds (316 allocations: 37.445 KiB)

----------------

total runtime:

----------------

0.254524 seconds (68.00 k allocations: 3.276 MiB)