CSCI 232:
Data Structures and Algorithms

Shortest Path (Part 2)

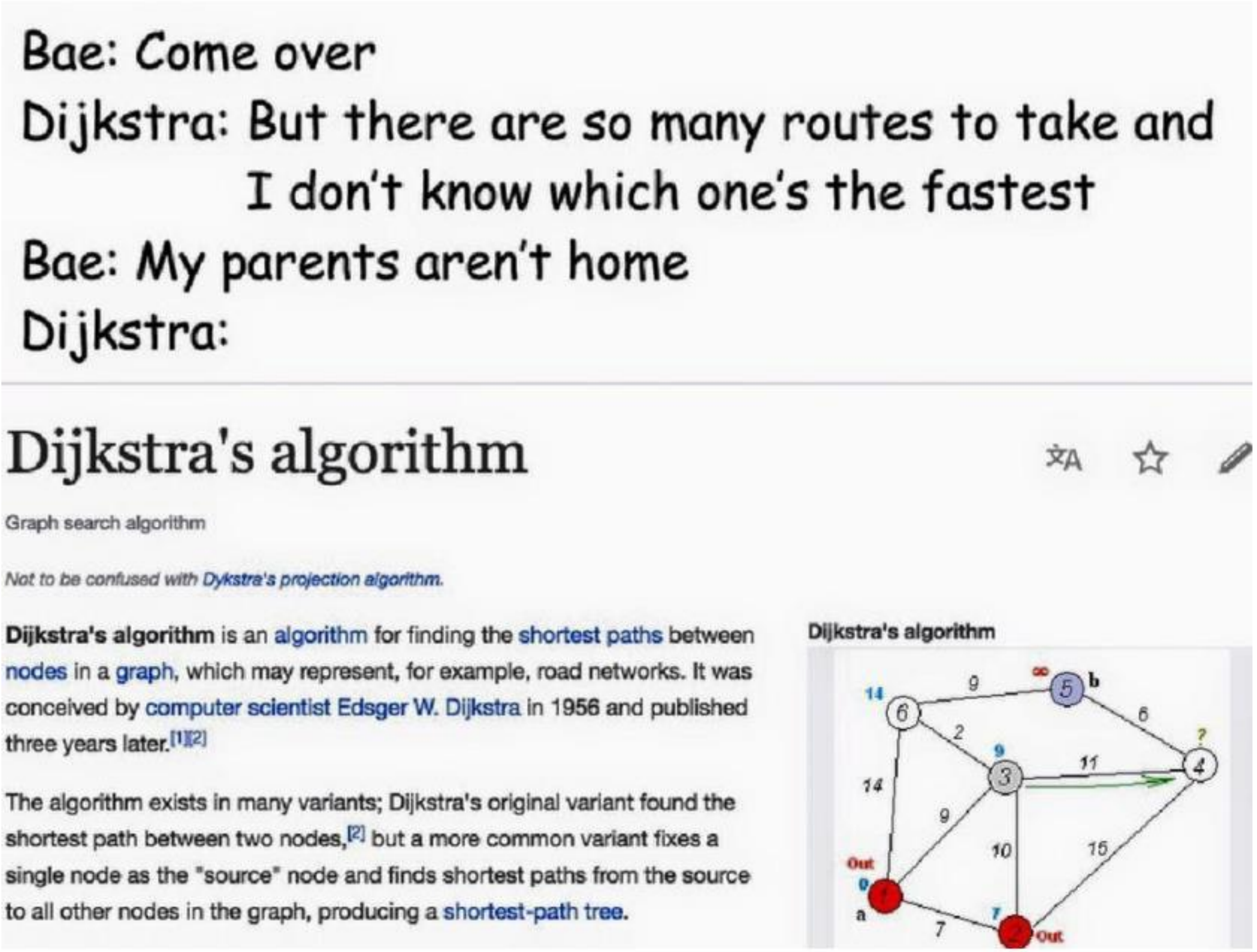
Reese Pearsall

Spring 2024

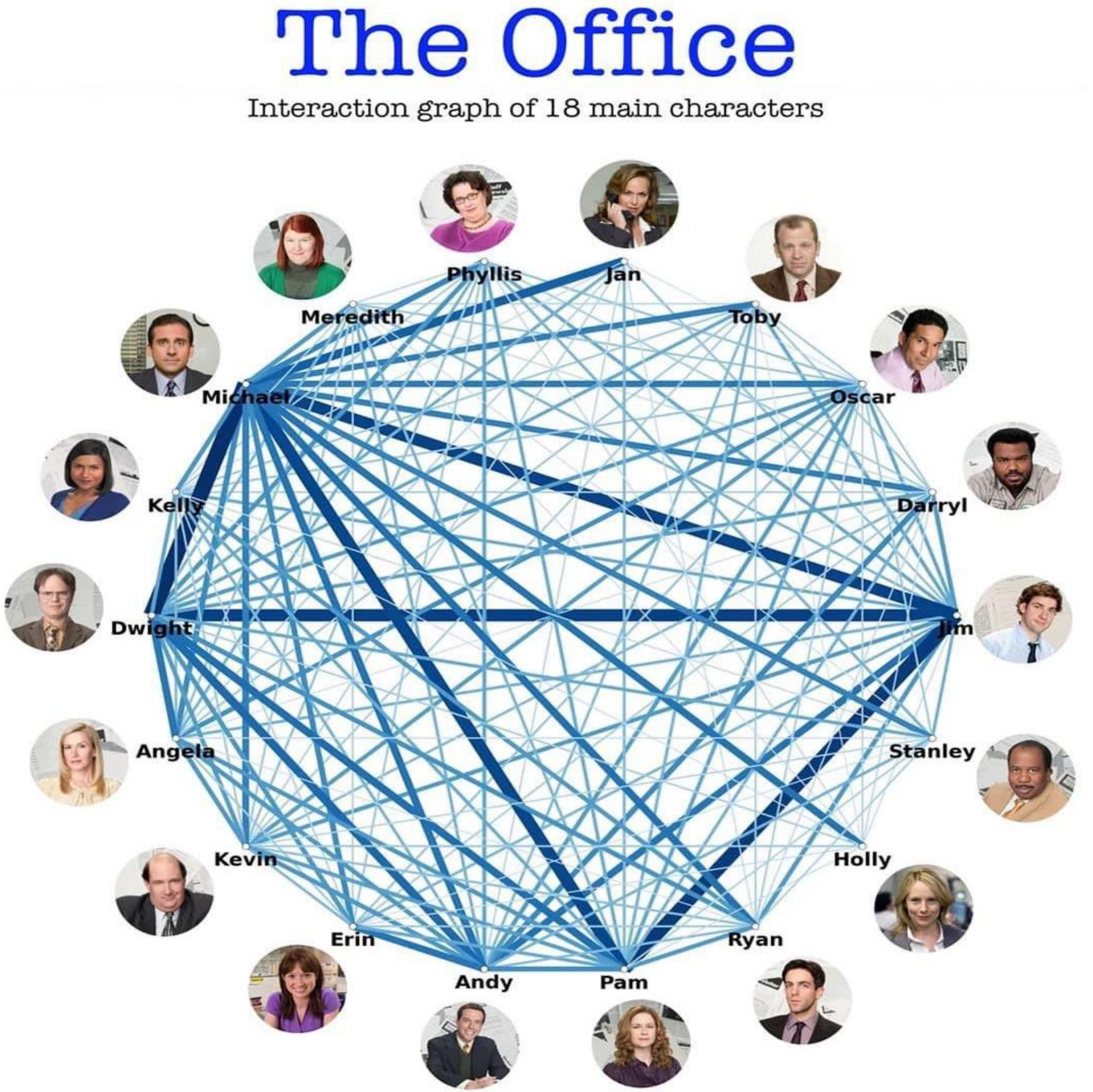
Announcements

Lab 10 due **Sunday at 11:59 PM** (Part 1 of Program 3)

Yes, there will be a program 4 (but it will be easier)

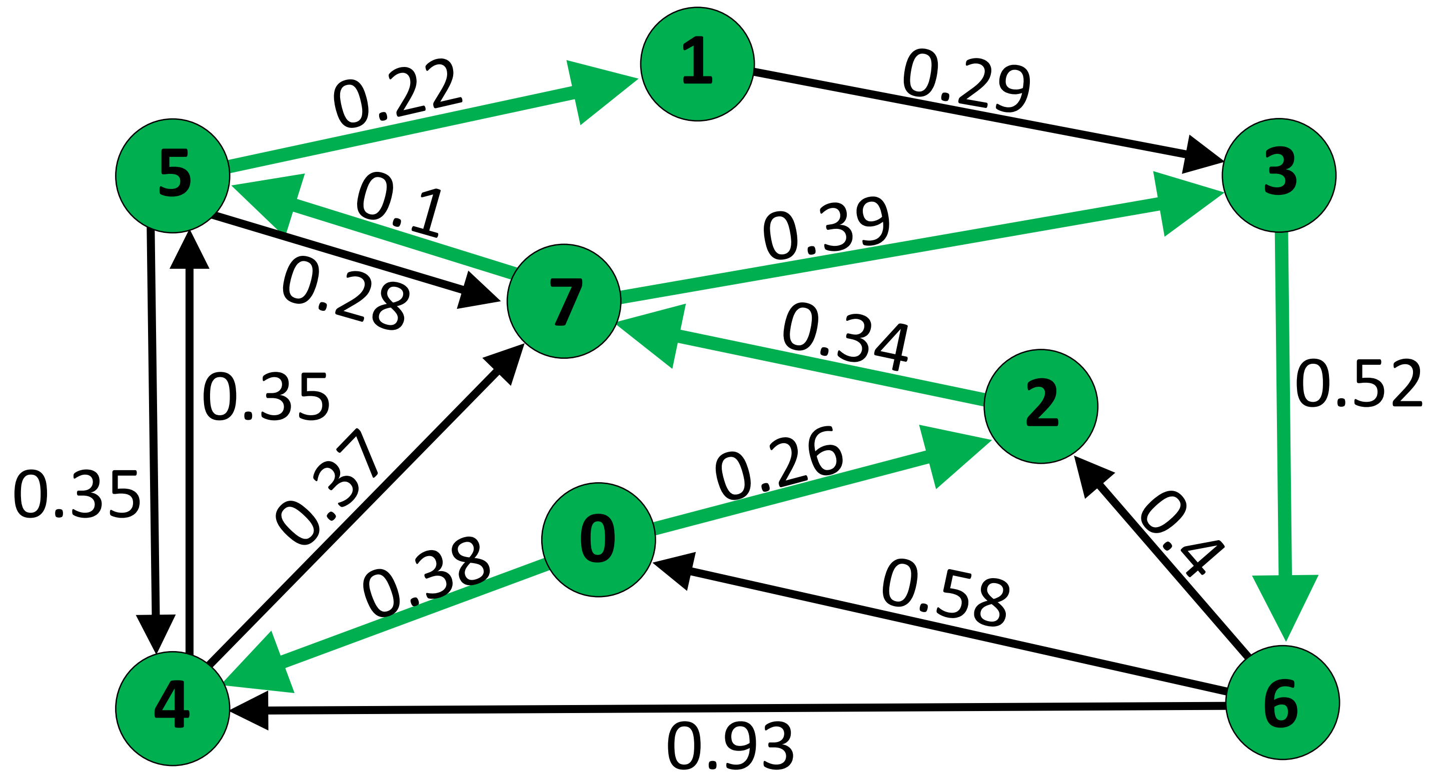


Lab 10


The Office
Interaction graph of 18 main characters
Phyllis
Jan
Meredith
Toby
Michael
Oscar
Kelly
Darryl
Dwight
Jim
Angela
Stanley
Kevin
Holly
Erin
Ryan
Andy
Pam

https://youtu.be/EFg3u_E6eHU

Dijkstra's Algorithm



Distance from 0

0	0
1	0.92
2	0.26
3	0.99
4	0.38
5	0.70
6	1.51
7	0.60

Previous vertex

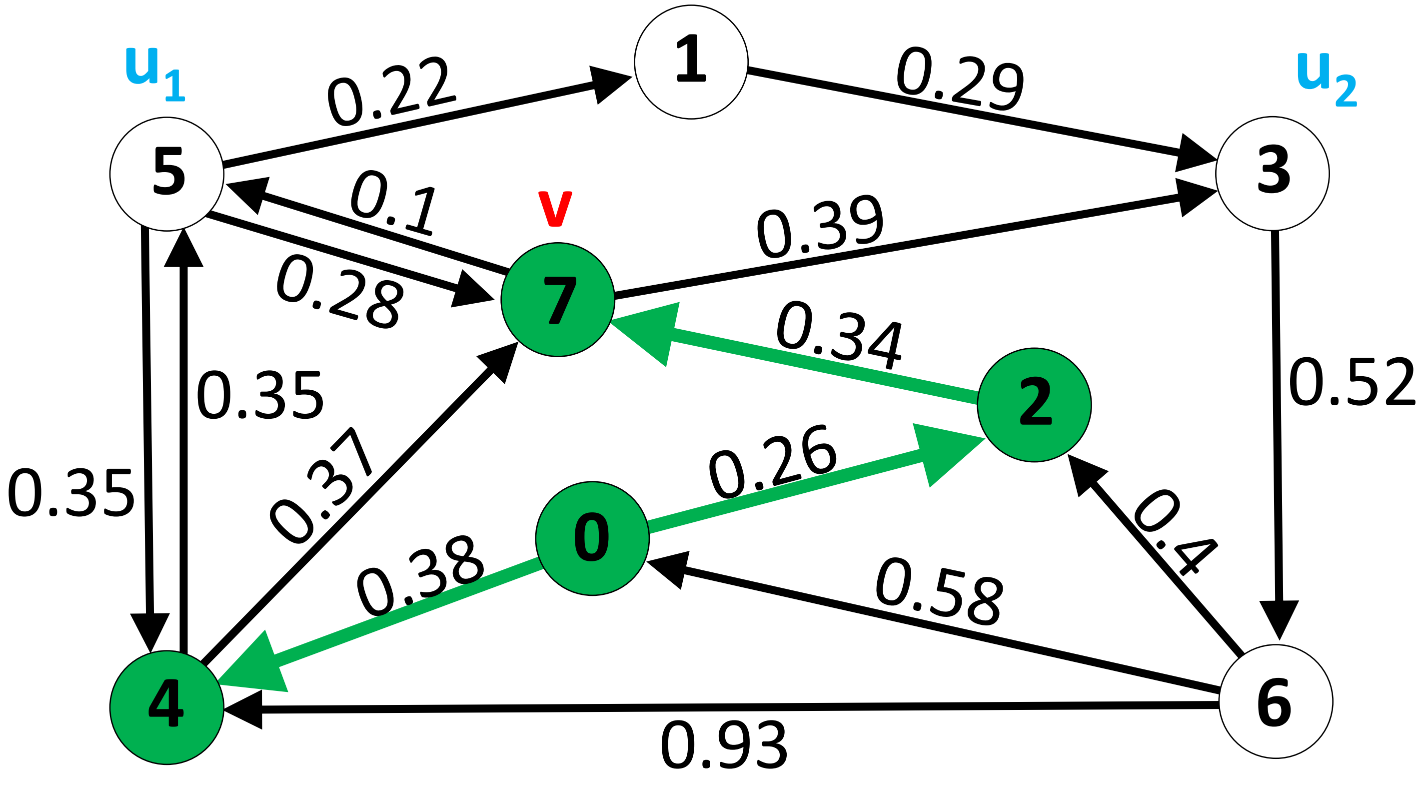
0	-
1	5
2	0
3	7
4	0
5	7
6	3
7	2

Priority queue

vertex (distance)

Rule: When processing vertex v, only add/modify queue for neighbor u if and only if:

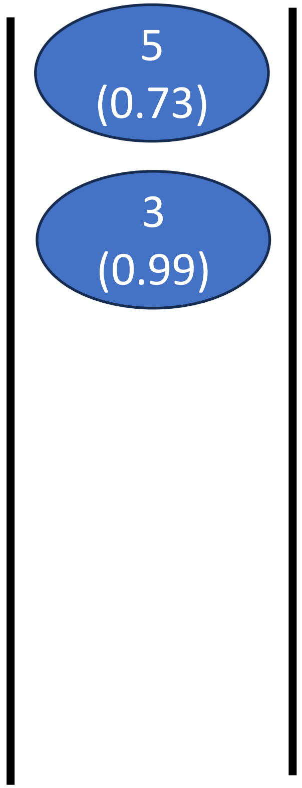
```
distance[v] + weight(v, u) < distance[u]
```



	Distance from 0
0	0
1	∞
2	0.26
3	0.99
4	0.38
5	0.73
6	∞
7	0.60

	Previous vertex
0	-
1	
2	0
3	7
4	0
5	4
6	
7	2

Priority queue



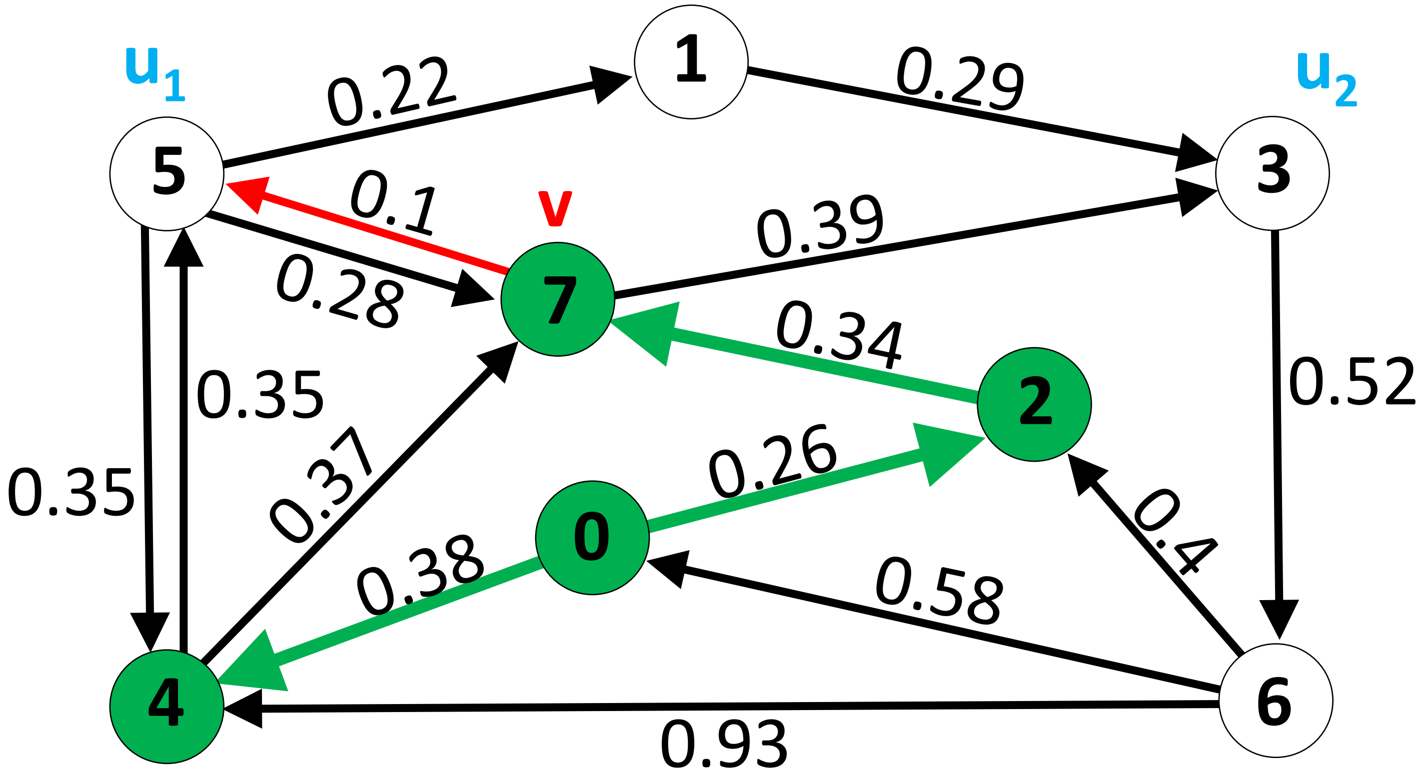
vertex (distance)

`PriortityVertex` Objects

Rule: When processing vertex v, only add/modify queue for neighbor u if and only if:

`distance[v] + weight(v, u) < distance[u]`

0.60 + 0.1 < 0.73



Vertex	Distance from 0		Previous vertex	
0	0		-	
1	∞			
2	0.26		0	
3	0.99		7	
4	0.38		0	
5	~~0.73~~	0.70	4	7
6	∞			
7	0.60		2	

Priority queue

- ~~5 (0.73)~~
- 3 (0.99)
- 5 (0.70)

vertex (distance)

`PriortityVertex` Objects

<u>Rule:</u> When processing vertex v, only add/modify queue for neighbor u if and only if:

`distance[v] + weight(v, u) < distance[u]`

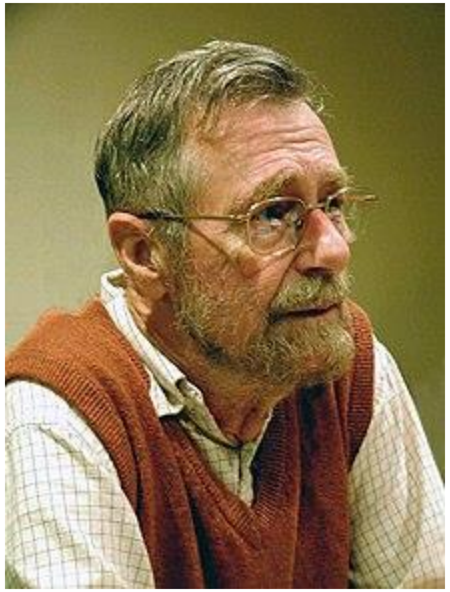
Dijkstra’s Algorithm

Running Time: **O(E · log(V))***

E = # of edges

V = # of vertices

* Varies depending on implementation and representation



Edsger Wybe Dijkstra
11 May 1930 – 6 August 2002

Proposition R. Dijkstra’s algorithm solves the single-source shortest-paths problem in edge-weighted digraphs with nonnegative weights.

Proof: If `v` is reachable from the source, every edge `v->w` is relaxed exactly once, when `v` is relaxed, leaving `distTo[w] <= distTo[v] + e.weight()`. This inequality holds until the algorithm completes, since `distTo[w]` can only decrease (any relaxation can only decrease a `distTo[]` value) and `distTo[v]` never changes (because edge weights are nonnegative and we choose the lowest `distTo[]` value at each step, no subsequent relaxation can set any `distTo[]` entry to a lower value than `distTo[v]`). Thus, after all vertices reachable from `s` have been added to the tree, the shortest-paths optimality conditions hold, and PROPOSITION P applies.

Proposition R (continued). Dijkstra’s algorithm uses extra space proportional to V and time proportional to $E \log V$ (in the worst case) to compute the SPT rooted at a given source in an edge-weighted digraph with E edges and V vertices.

Proof: Same as for Prim’s algorithm (see PROPOSITION N).

Proposition N (continued). Kruskal’s algorithm uses space proportional to E and time proportional to $E \log E$ (in the worst case) to compute the MST of an edge-weighted connected graph with E edges and V vertices.

Proof: The implementation uses the priority-queue constructor that initializes the priority queue with all the edges, at a cost of at most E compares (see SECTION 2.4). After the priority queue is built, the argument is the same as for Prim’s algorithm. The number of edges on the priority queue is at most E, which gives the space bound, and the cost per operation is at most $2 \lg E$ compares, which gives the time bound. Kruskal’s algorithm also performs up to E `find()` and V `union()` operations, but that cost does not contribute to the $E \log E$ order of growth of the total running time (see SECTION 1.5).

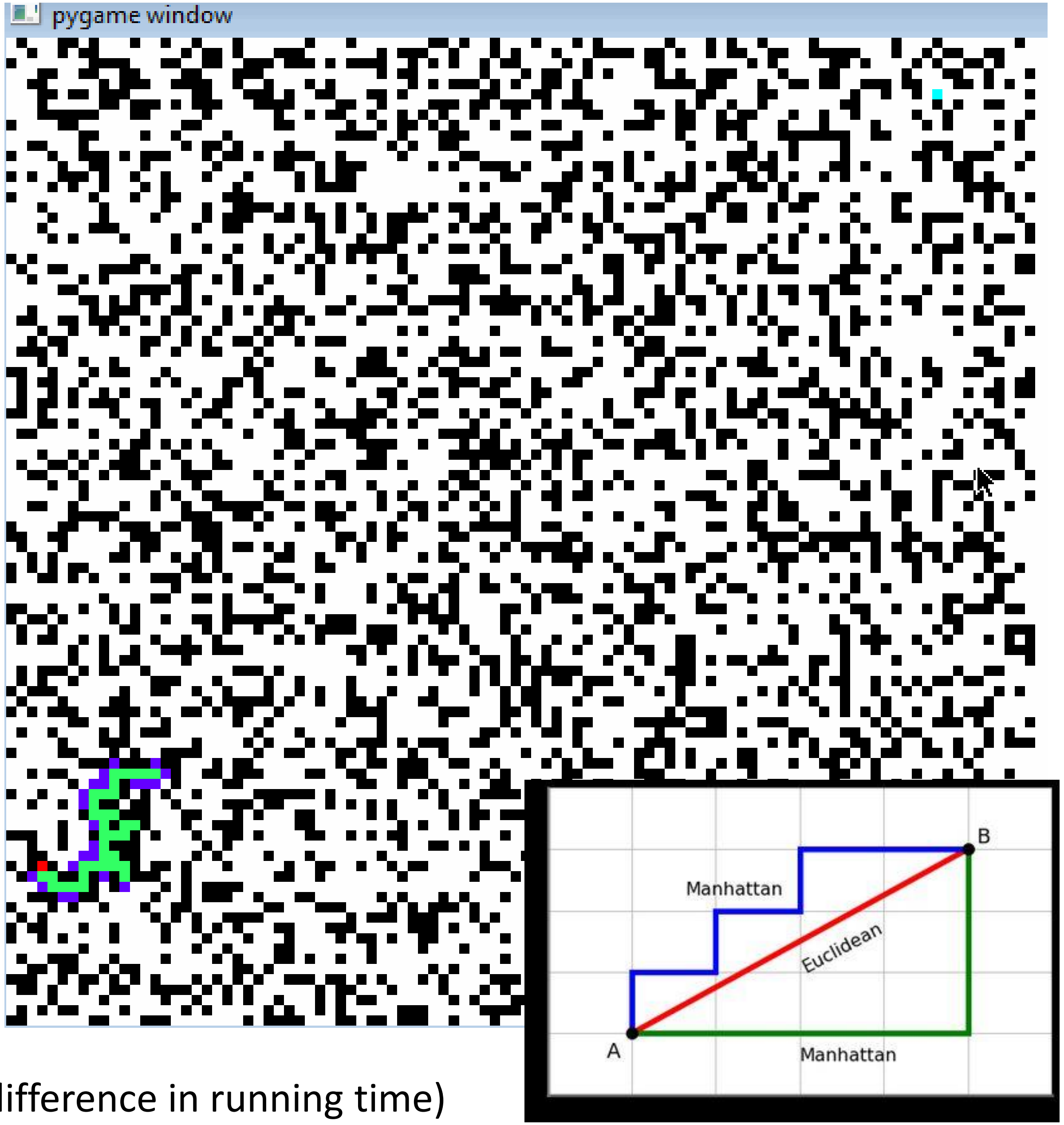
A Star

A Star or **A*** is another algorithm that will compute the shortest path in a graph

In **Dijkstra's Algorithm** we select the least-cost unvisited node, and we compute the shortest path to all other nodes

In **A***, we select the node that is the shortest distance away from the target, and does not compute the shortest path to all other nodes

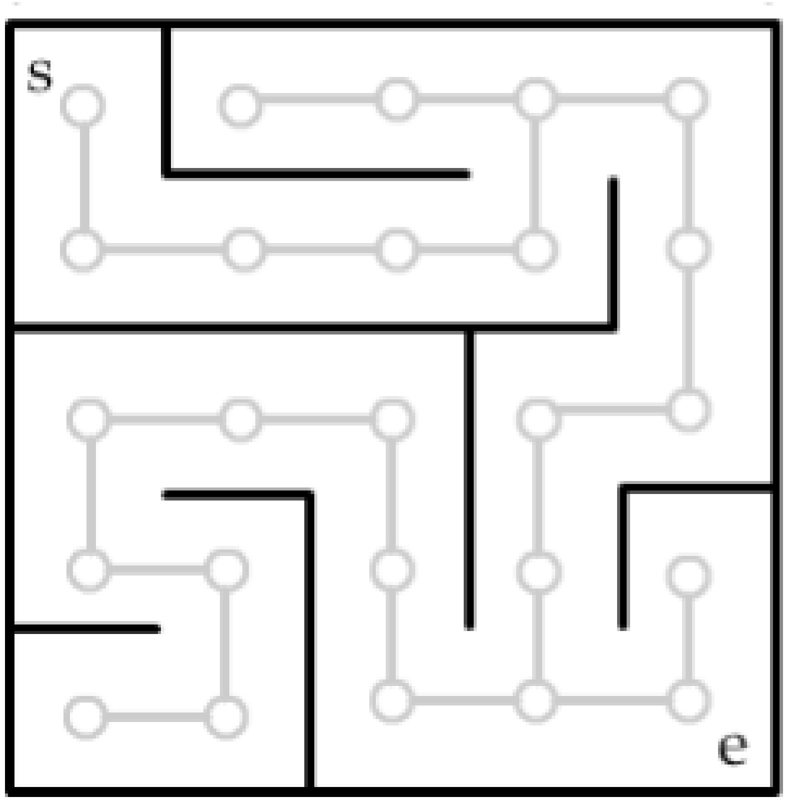
In A* we use a **heuristic** to make decisions

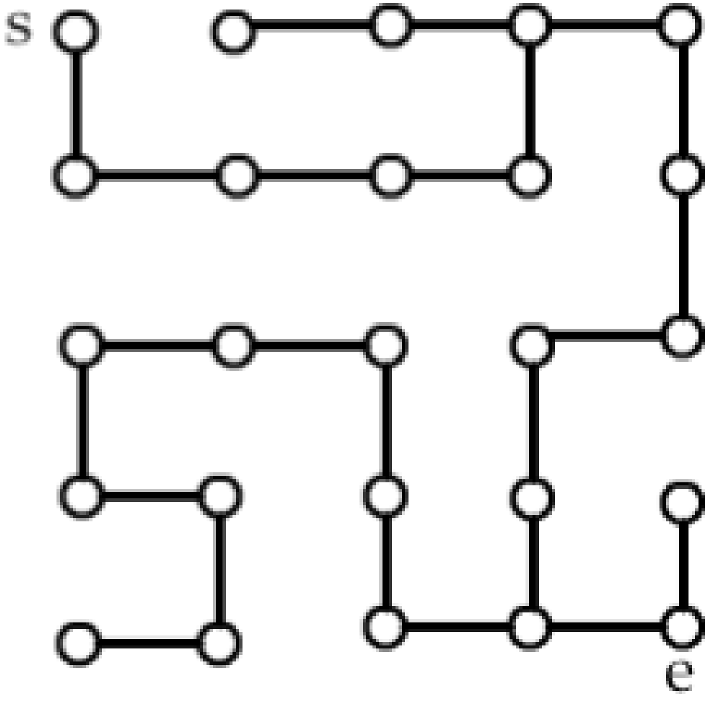


Euclidean Distance heuristic

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

(No difference in running time)





We can represent a maze using graphs!

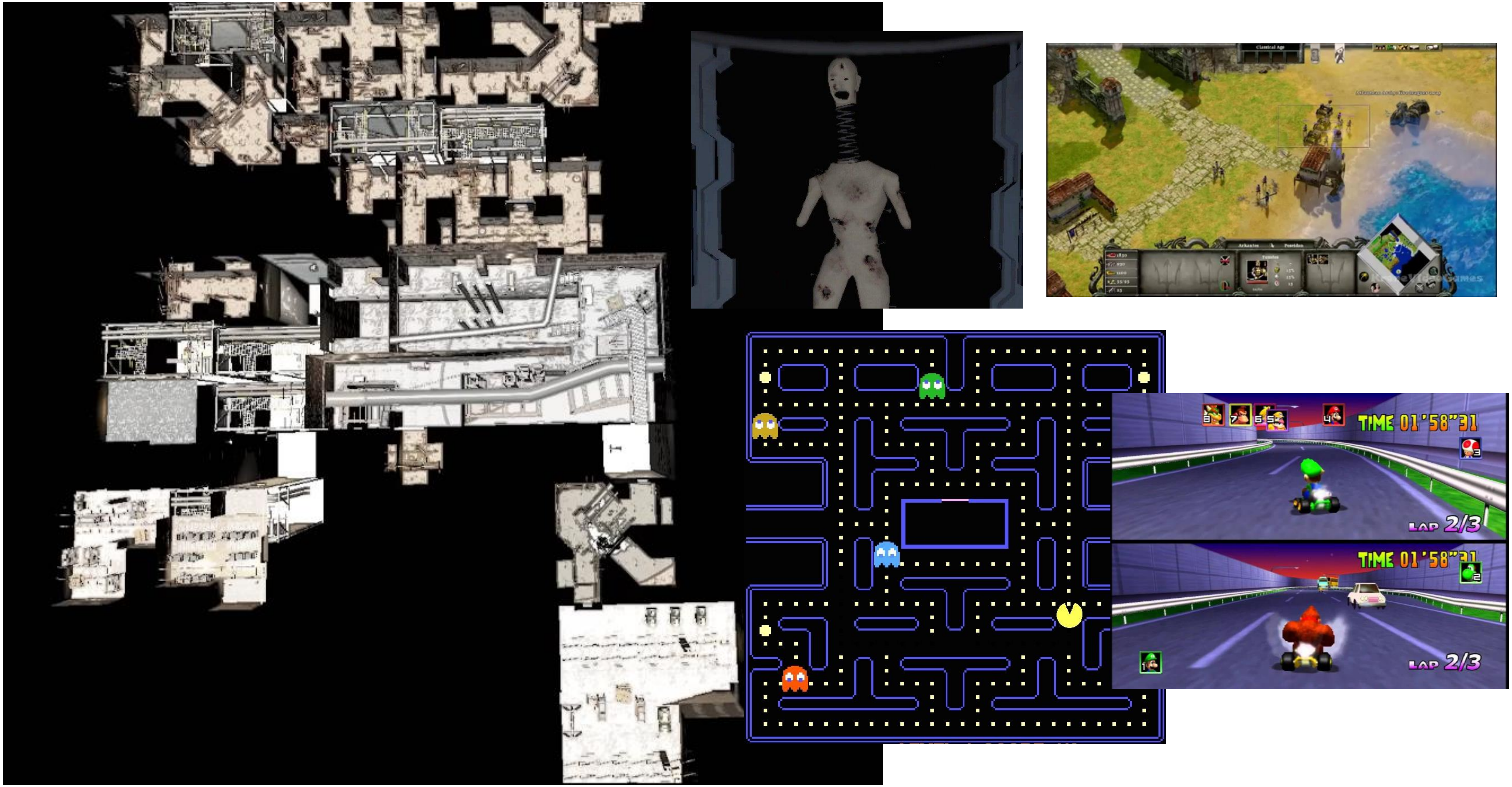
Creating Mazes with Depth First

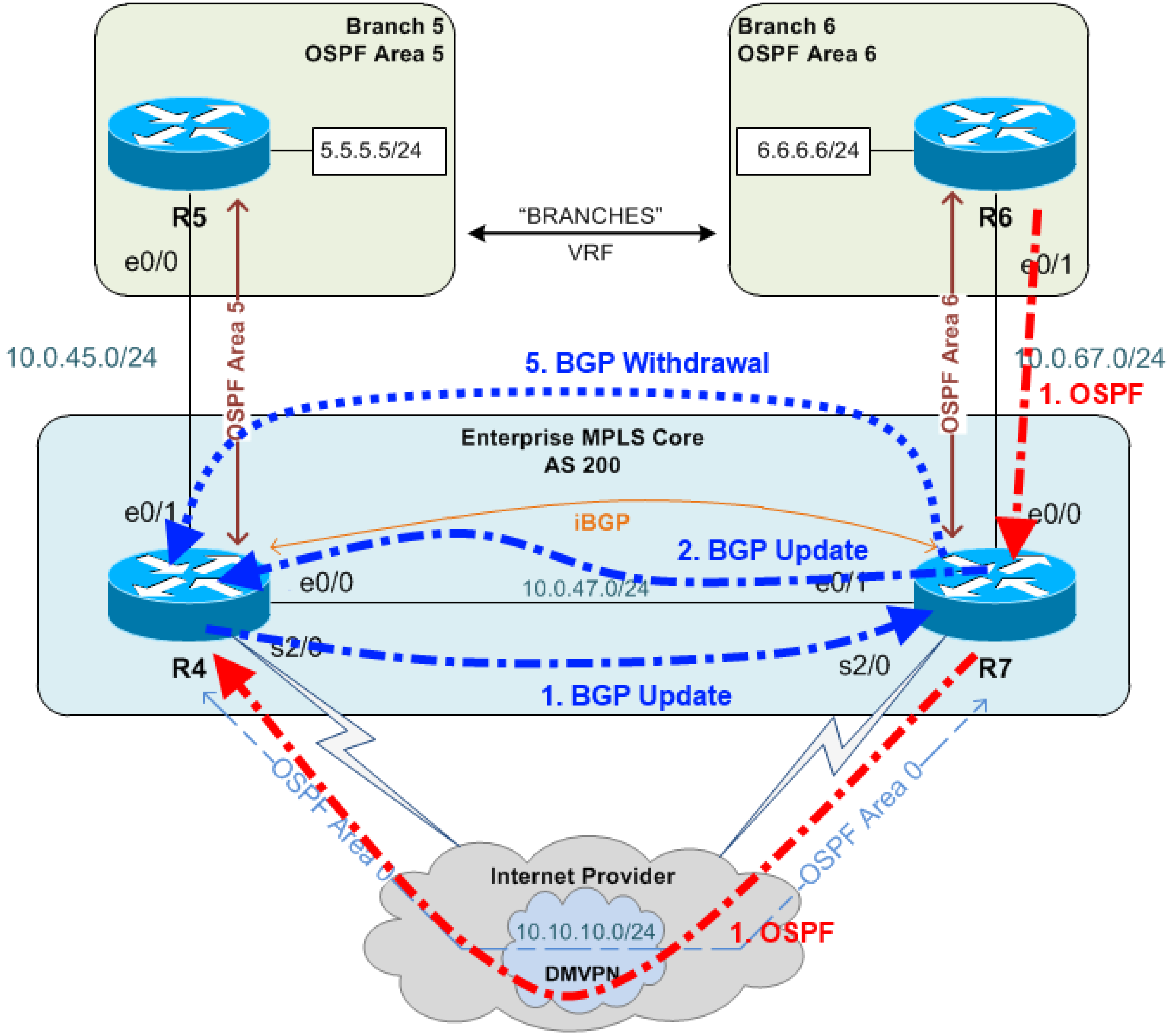
https://www.youtube.com/watch?v=e5zDG4JIsyg

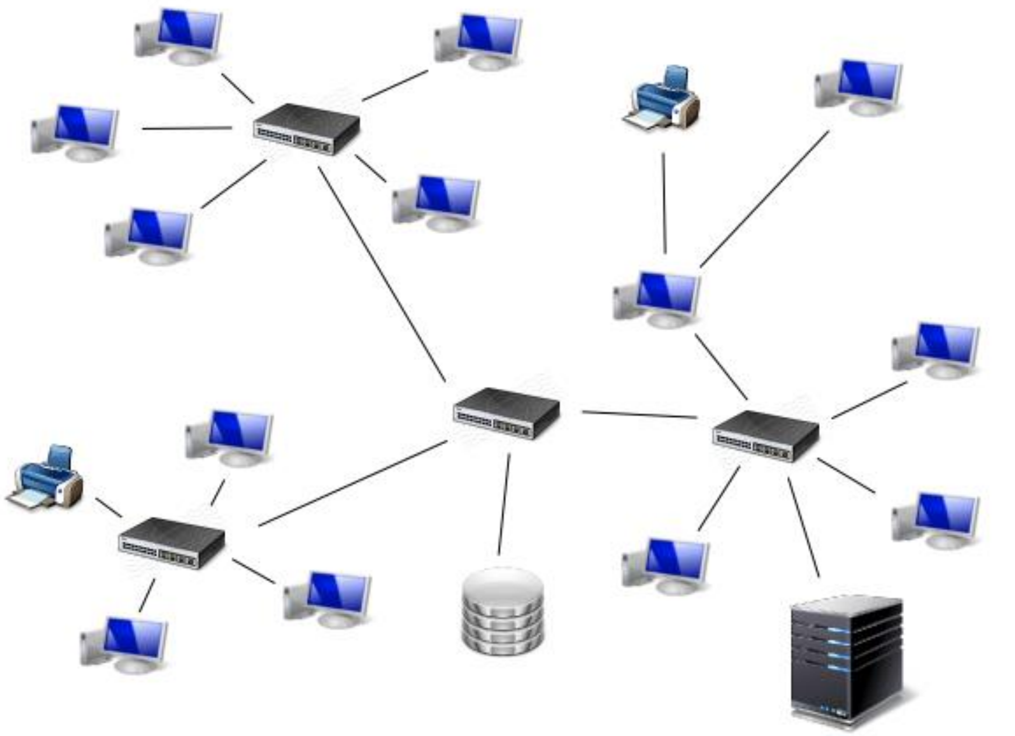
Shortest Path Algorithms on Maze

https://clementmihailescu.github.io/Pathfinding-Visualizer/

Applications of Shortest Path?

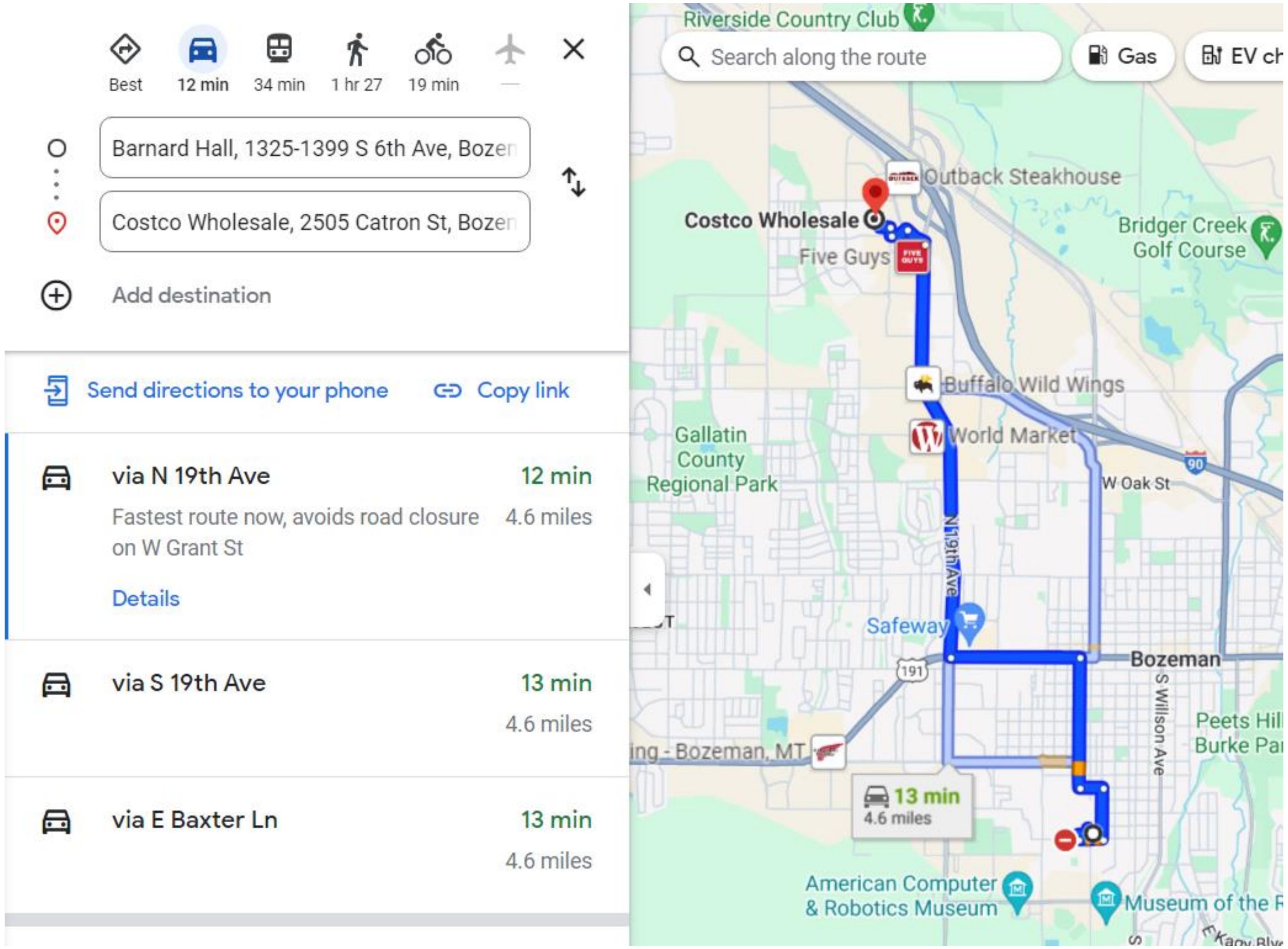

TIME 01'58"31
LAP 2/3
LAP 2/3





Dijkstra's Algorithm is used for network routing

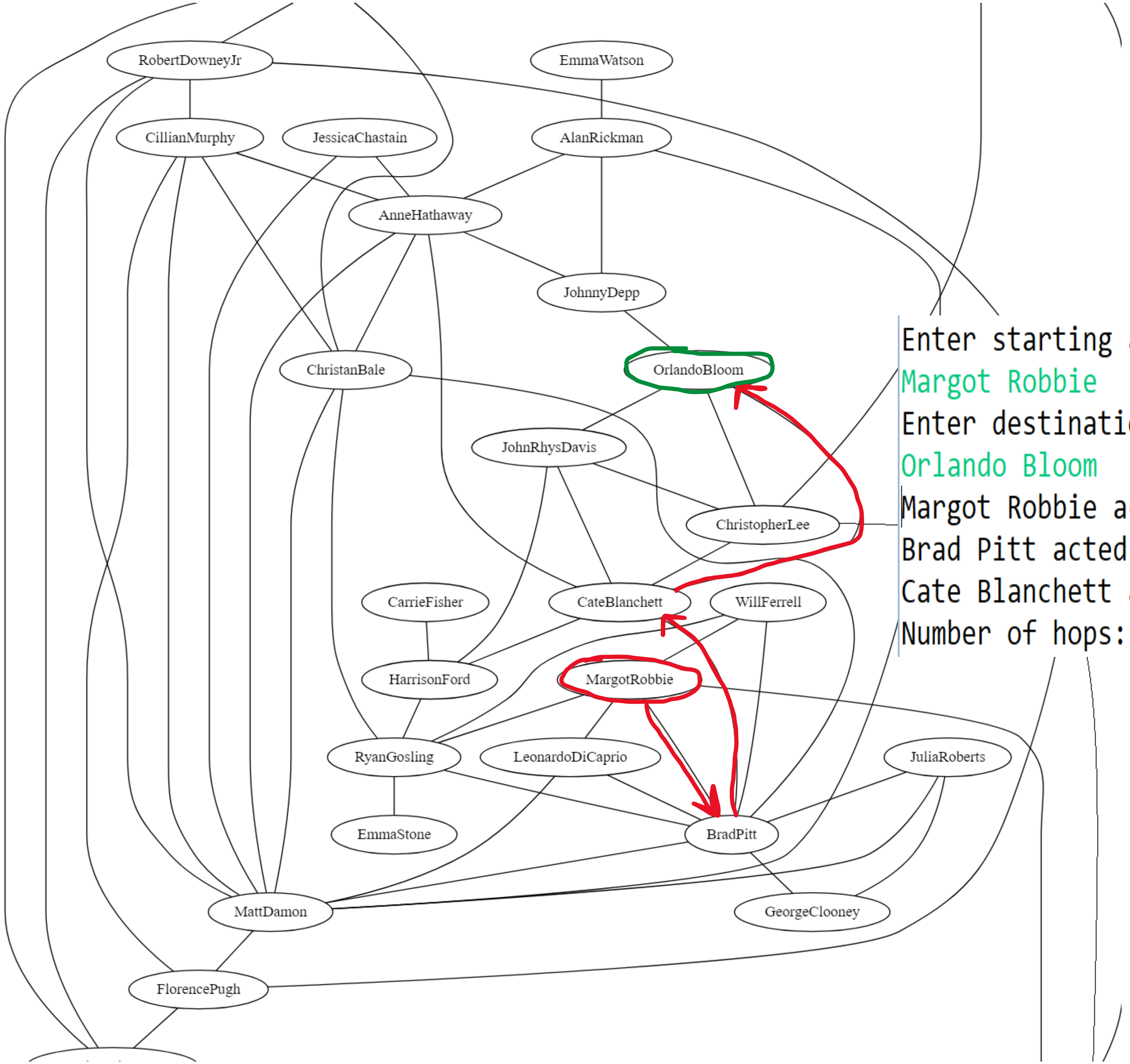
The **OSPF** Protocol



Finding shortest path on a map

Sending drones or robots on the shortest path

Finding Shortest Path between actors




```
Enter starting actor:
Margot Robbie
Enter destination actor:
Orlando Bloom
Margot Robbie acted with Brad Pitt in Once Upon a Time in Hollywood
Brad Pitt acted with Cate Blanchett in The Curious Case of Benjamin Button
Cate Blanchett acted with Orlando Bloom in Fellowship of the Ring
Number of hops: 3
```

Oracle of Bacon

Program 3

* And MST tree

Readability and Coding Style