Graph and Network Analysis

Introduction

- Problem Discussion
- Target

Methodology and Tool

- NetworkX
- Algorithm

Experiments

- Prove that Q-Learning can find the shortest path
- Compare the computing speed between A* , Dijkstra and Q-learning algorithm

Conclusion

Introduction

Problem Discussion

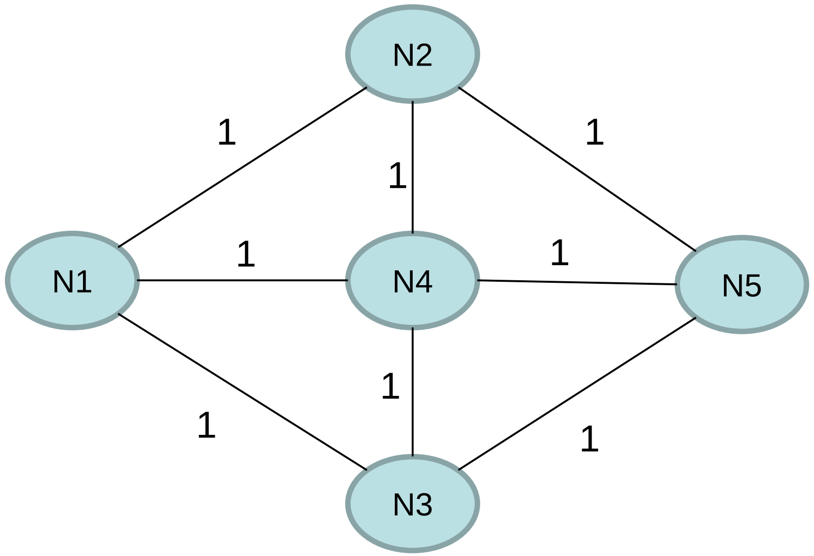
- Assume that the problem is based on graph. Given a graph $G = (V, E)$, V is a non-emptyfinite set of **nodes**. E is a non-empty finite set of **edges**, which are 2-element subsets of V
- **Static / Dynamic** means the **attributes of edges** (weight, congestion…) are **stable / unstable**
- **Known Topology :** $G = (V, E)$ structure is **full observable**, which means each node knows all others nodes / edges in $G = (V, E)$
- **Unknown Topology :** $G = (V, E)$ structure is **partially observable,** which means each node can only observe the structure of a **local area**.

Target

- To find the shortest path in { **Static , Dynamic }** x { **Known Topology , Unknown Topology }** environment.

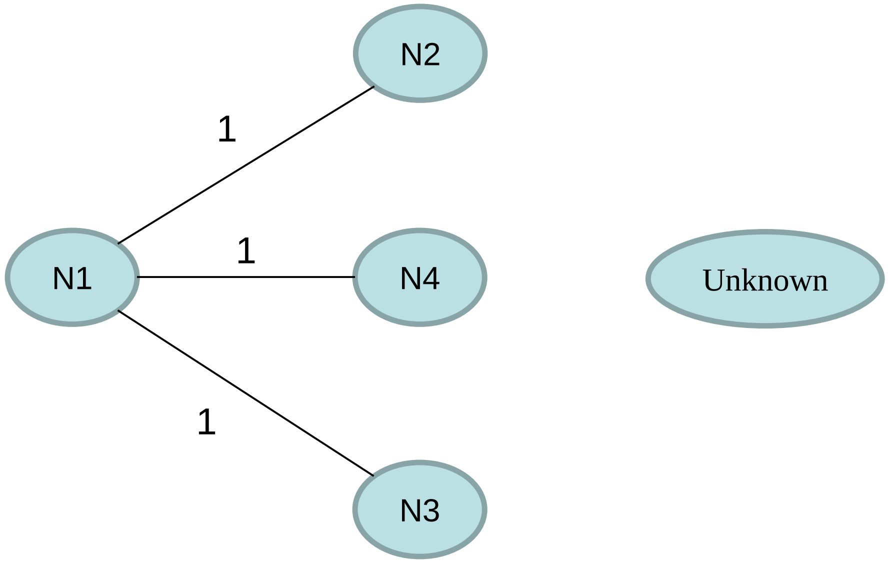
Introduction

A. Static + Known Topology



For **N1**, it knows all information of $G = (V, E)$, such as each node's location and each edge's attributes (e.g., weight, distance…).

B. Static + Unknown Topology

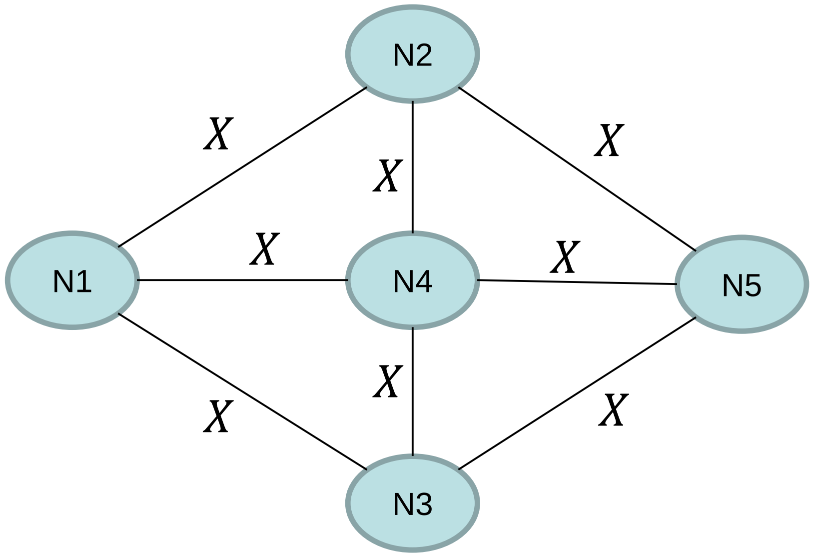


For **N1**, it only observes its neighbors' location and knows the weights of edges.

Mobile Communications Networking Lab, NTU

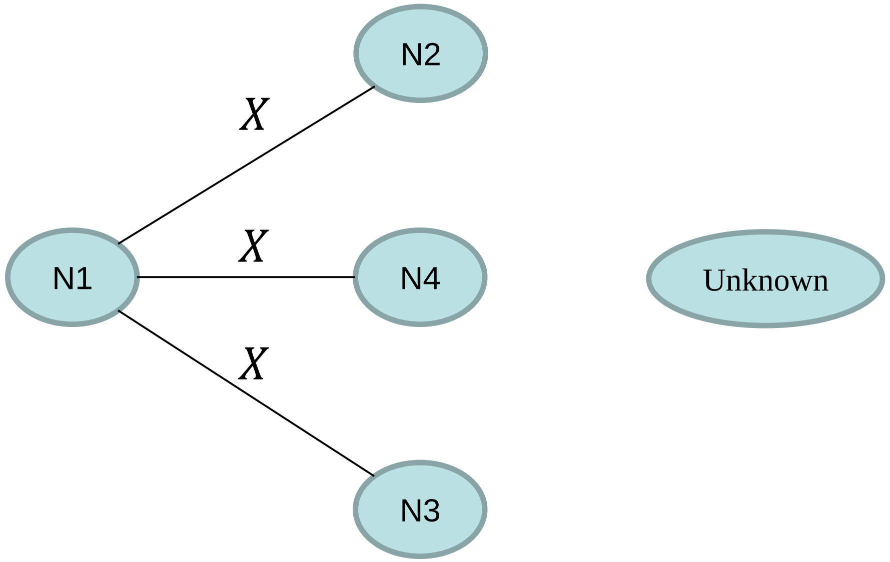
Introduction

C. Dynamic + Known Topology



For **N1**, it knows some information of $G = (V, E)$, such as each node's location. But the attributes (e.g., weight, distance…) of each edge are dynamic, according to a R.V. X

D. Dynamic + Unknown Topology



For **N1**, it only observes its neighbors' location. And the attributes (e.g., weight, distance…) of each edge are dynamic, according to a R.V. X

Mobile Communications Networking Lab, NTU

Methodology and Tool

NetworkX

NetworkX is a Python package for the creation, manipulation of the structure, dynamics, and functions of complex networks.

- Data structures for graphs, digraphs, and multigraphs
- Standard graph algorithms
- Network structure and analysis measures
- Generators for classic graphs, random graphs, and synthetic networks
- Nodes can be "anything" (e.g., text, images…)
- Edges can hold arbitrary data (e.g., weights, time-series)

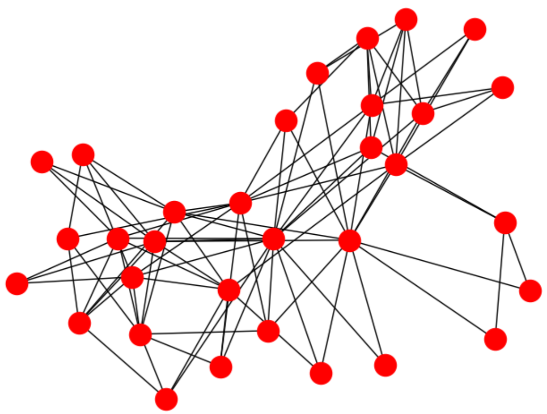
Methodology and Tool

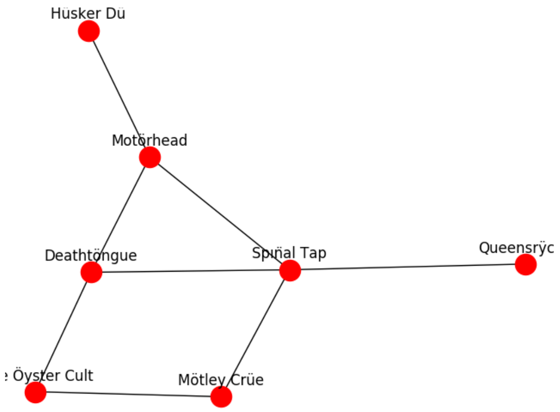
References

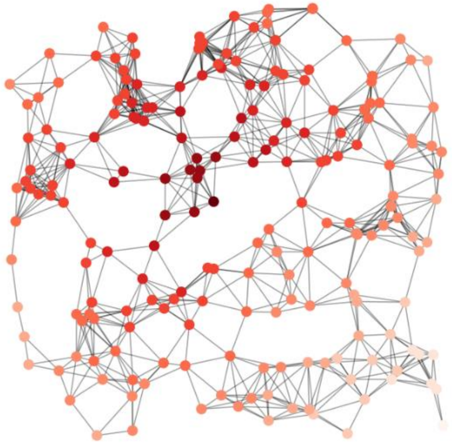
https://link.springer.com/content/pdf/10.1007%2F978-3-319-53004-8.pdf

https://networkx.github.io/documentation/stable/tutorial.html

https://github.com/AvisChiu/Networkx_Graph







Methodology and Tool

Algorithm

- **Dijkstra algorithm:** Dijkstra explores lower cost paths instead of exploring all possible paths equally. Dijkstra’s Algorithm works well to find the shortest path, but it wastes time exploring in directions that aren’t promising.

- **A* :** A* is a popular choice for graph search. It is a modification of Dijkstra’s Algorithm that is optimized for a single destination. Dijkstra’s Algorithm can find paths to all locations; A* finds paths to one location, or the closest of several locations. It prioritizes paths that seem to be leading closer to a goal.

- **Difference:** Dijkstra’s Algorithm works well to find the shortest path, but it **wastes time exploring in directions that aren’t promising**. The A* algorithm uses *both* the **actual distance** from the start and the **estimated distance** to the goal.

https://www.redblobgames.com/pathfinding/a-star/introduction.html

Experiments

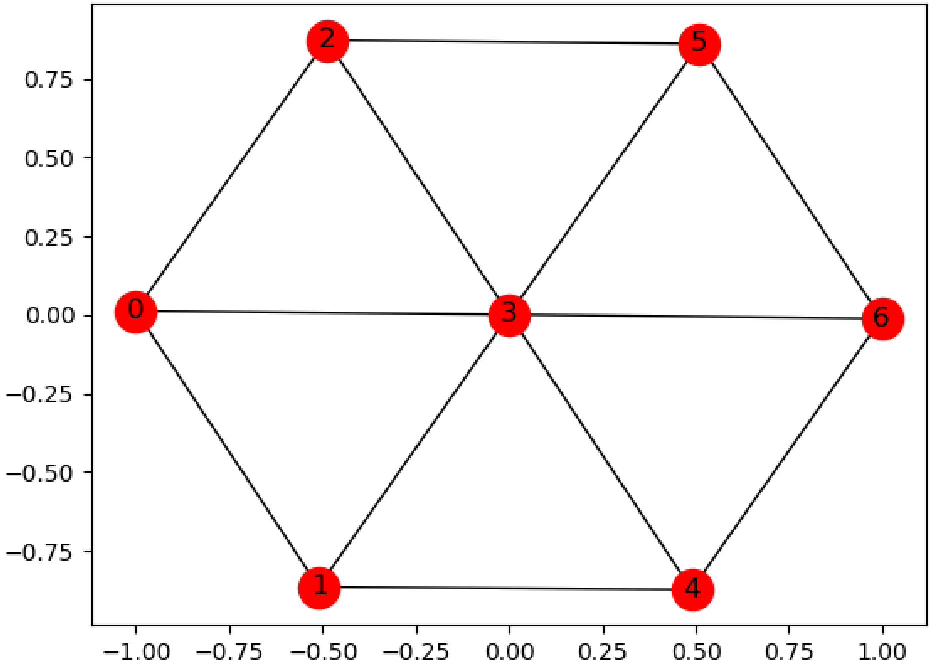


Fig.1 Static + Known Topology

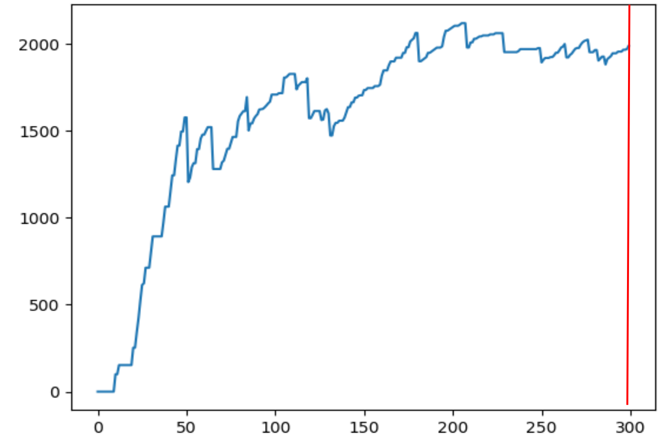
Using NetworkX to make a graph (Fig.1)

```
data = [(0, 1, 1), (0, 2, 2), (0, 3, 3),
        (1, 0, 1),(1, 3, 2),(1, 4, 3),
        (2, 0, 2),(2, 3, 1),(2, 5, 4),
        (3, 0, 3),(3, 1, 2),(3, 2, 1),(3, 4, 1),
        (4, 1, 3),(4, 3, 1),(4, 6, 2),
        (5, 2, 4),(5, 3, 1),(5, 6, 2),
        (6, 3, 1),(6, 4, 2),(6, 5, 2)]
```

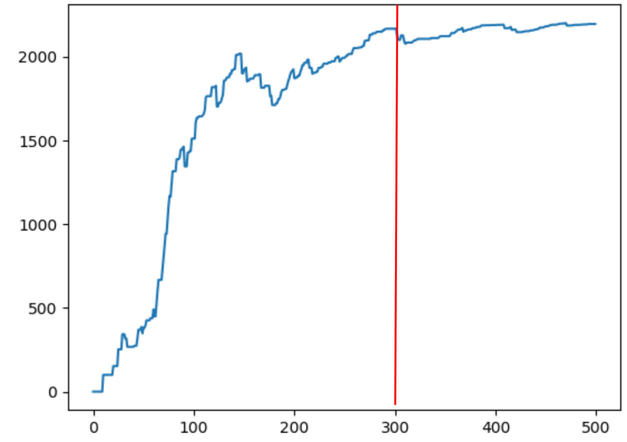
Mobile Communications Networking Lab, NTU

Experiments (Q-Learning)

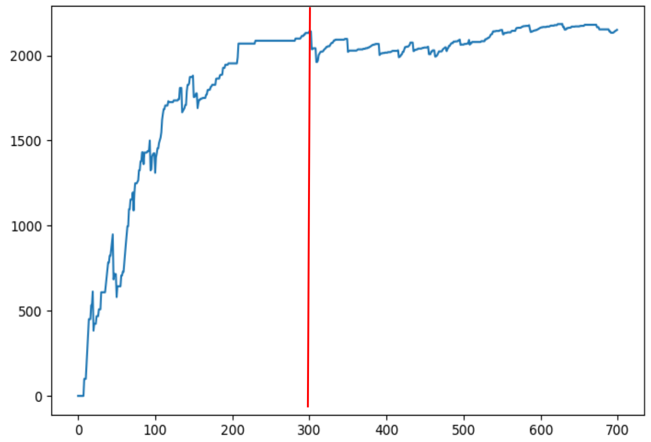
Episode：300



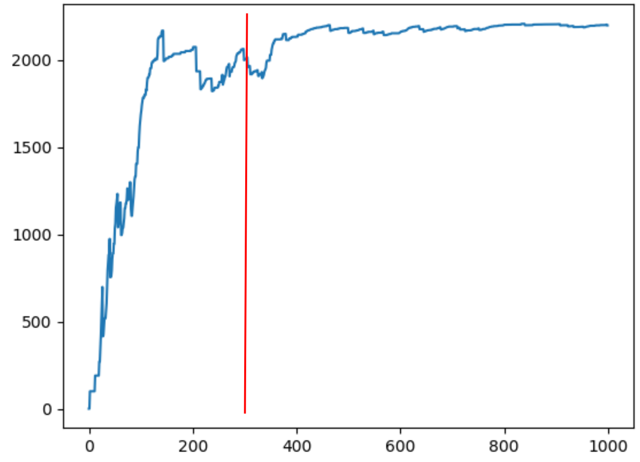
Episode：500



Episode：700



Episode：1000



Conclusion

Q-Learning is proved that it can be used to find the shortest path in **Static + Known Topology** environment.

But its computing speed is much slower than **A Star** and **Dijkstra**, shown in Table 1.

Average Operation Time means the time cost of find the shortest path, which can represent the performance of algorithm. By using Q-Learning, it takes about 500 iterations to converge to a solution, but the shortest path can be got at about 300^{th} iteration, even though the Q-Table still in diverge.

	Average Operation Time
A Star	**0.000159 s**
Dijkstra	**0.000125 s**
Q-Learning	**0.080330 s**

Table 1. Average Operation Time Comparison

Mobile Communications Networking Lab, NTU

Conclusion

will be soon

	Static + Known Topology	Static + Unknown Topology	Dynamic + Known Topology	Dynamic + Unknown Topology
A Star	Fast	?	?	?
Dijkstra	Fast	?	?	?
Q-Learning	Slow	?	?	?

? = waiting to be proved

End