Comparing Pathfinding Agents: Heuristic vs. Reinforcement Learning

Sohan Mekala

Introduction

Graph-related problems are widespread in the field of computer science and have applications in various fields, including network optimization, logistics, and resource allocation. These problems often involve finding optimal paths or routes through complex systems represented as graphs, where both nodes and edges can carry significant meaning (Cormen et al. 123). Potential applications of such graph-based optimization include:

- Network routing in telecommunications
- Supply chain optimization
- Urban planning and traffic management
- Resource allocation in distributed computing systems

While many traditional algorithms focus on finding shortest paths based solely on edge weights, there exists a gap in addressing scenarios where node weights are equally important. This gap is particularly evident in situations where computing resources or capacities at intermediate points (represented by nodes) play a crucial role in determining the feasibility and efficiency of a path (Schrijver 45).

Dijkstra's algorithm, developed by computer scientist Edsger W. Dijkstra in 1956, has long been a cornerstone in solving shortest path problems. Its elegance lies in its ability to efficiently find the shortest path between nodes in a graph, considering edge weights (Dijkstra 269). However, Dijkstra's algorithm has limitations, particularly in scenarios where node values are as critical as edge weights. It doesn't inherently account for the "cost" or "capacity" associated with visiting nodes, which can be a significant factor in many real-world applications.

To address these limitations and explore more adaptive approaches, this study introduces a comparison between a heuristic method based on Dijkstra's algorithm and a reinforcement learning (RL) approach using Q-learning. Reinforcement learning, a branch of machine learning, allows agents to learn optimal strategies through interaction with an environment. Q-learning, in particular, is a model-free RL technique that can learn to make decisions in sequential decision-making problems, potentially adapting to complex constraints involving both edge and node weights (Sutton and Barto 131).

This research aims to compare the effectiveness of these two approaches in solving path planning problems where both edge weights (representing time or distance) and node weights (representing computing resources or capacities) are considered. By doing so, I hope to shed light on the strengths and weaknesses of each method and provide insights into their applicability in various scenarios.

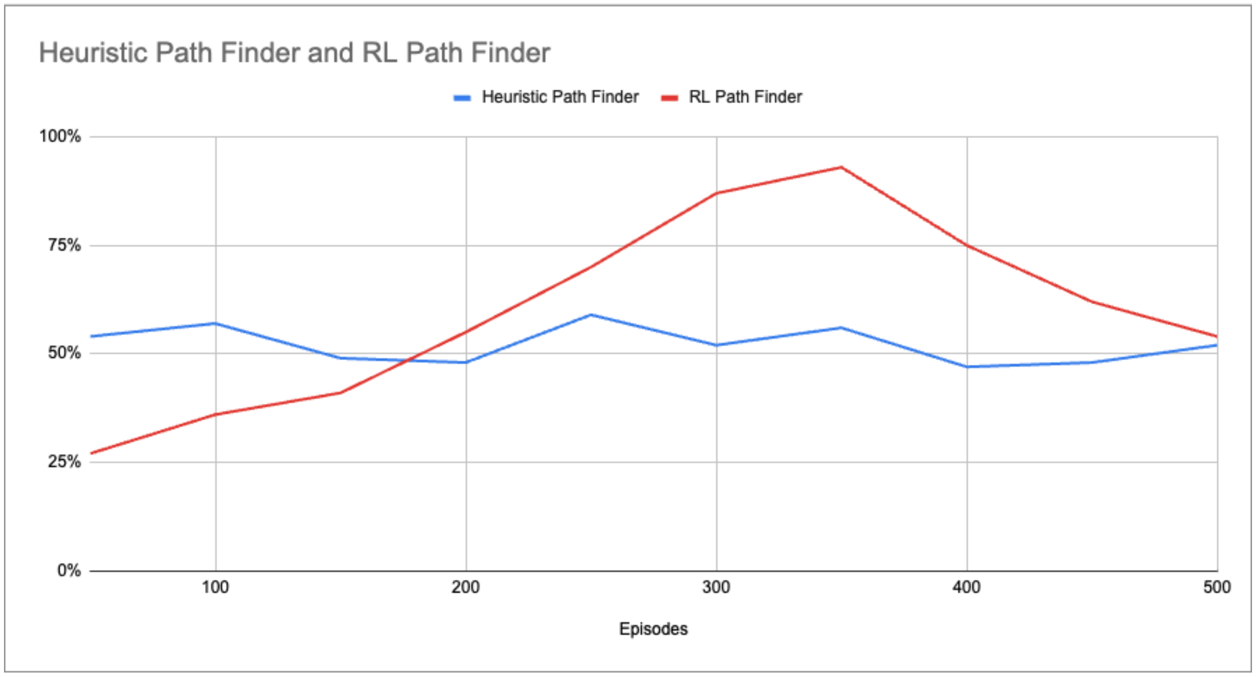
Methodology:

This research involves comparing two pathfinding approaches: a heuristic-based method using Dijkstra's algorithm and a reinforcement learning-based method employing Q-learning. The experimental setup includes the following steps:

1. **Graph Generation**: A large graph with 20 nodes and randomly generated edges is created. Node values represent computing resources, and edge values represent wait times. The graph is ensured to be connected with additional random edges.
2. **Request Generation**: Random requests are generated with a start node, end node, and required computing resources.
3. **Heuristic Approach**: The heuristic approach utilizes Dijkstra's algorithm to find paths based on edge weights. The algorithm is adapted to ensure paths meet computing resource requirements and do not exceed maximum duration constraints.
4. **Reinforcement Learning Approach**: A Q-learning-based pathfinding agent is implemented. The agent learns optimal paths through exploration and exploitation, updating a Q-table based on rewards and penalties related to resource availability and path duration.
5. **Evaluation**: Both approaches are tested against a set of requests, and success rates are compared based on the proportion of successful requests to the total number of requests.

*Code can be seen in this link:
https://github.com/SohanMekala/Pathfinder-Agent-Research/blob/main/agent_comparision.py*

Results:



The comparison between the Heuristic Path Finder and the Reinforcement Learning (RL) Path Finder reveals distinct performance patterns across varying episode counts. The Heuristic approach demonstrates consistent performance, maintaining a success rate of around 50% regardless of the number of episodes. This stability suggests a reliable, albeit limited, effectiveness in path-finding tasks.

In contrast, the RL Path Finder shows a more dynamic performance profile. Starting with a lower success rate of about 25% at 100 episodes, it rapidly improves, surpassing the heuristic method at approximately 150-200 episodes. The RL approach reaches peak performance near 350 episodes, achieving a success rate close to 95%. However, beyond this point, its performance declines, dropping to about 50% at 500 episodes.

This performance curve highlights several key insights. First, the RL method demonstrates superior adaptability, significantly outperforming the heuristic approach at its peak. However, its effectiveness is highly dependent on the number of learning episodes. The decline in performance after 350 episodes suggests potential overfitting or learning instabilities when training is extended too far.

These findings indicate that while the RL approach can offer superior performance, it requires careful tuning to achieve optimal results. The heuristic method, although less effective at its peak, provides consistent performance that may be preferable in scenarios where reliability is prioritized over maximum efficiency.

Conclusion:

This study compared the performance of a Heuristic Path Finder and a Reinforcement Learning (RL) Path Finder in graph-based path planning tasks. Our findings reveal that while the Heuristic approach offers consistent performance regardless of episode count, the RL method demonstrates superior adaptability and peak performance when properly tuned. The RL Path Finder's effectiveness, however, is heavily dependent on the number of training episodes, with performance declining after an optimal point.

These results highlight the trade-offs between consistency and peak performance in path-finding algorithms. The choice between heuristic and RL approaches should be guided by specific application requirements, considering factors such as the need for reliability versus maximum efficiency, and available computational resources for training.

Future research could explore techniques to maintain RL performance at higher episode counts and investigate the applicability of these findings to more complex graph structures and real-world scenarios.

Works Cited

Cormen, Thomas H., et al. Introduction to Algorithms. 3rd ed., MIT Press, 2009.

Dijkstra, E. W. "A Note on Two Problems in Connexion with Graphs." Numerische Mathematik, vol. 1, no. 1, 1959, pp. 269-271.

Schrijver, Alexander. Combinatorial Optimization: Polyhedra and Efficiency. Springer, 2003.

Sutton, Richard S., and Andrew G. Barto. Reinforcement Learning: An Introduction. 2nd ed., MIT Press, 2018.