Faster Shortest Path Computation for Traffic Assignment

THE UNIVERSITY OF AUCKLAND
FACULTY OF ENGINEERING

Boshen Chen Department of Engineering Science

Supervisors: Dr. Andrea Raith and Olga Perederieieva

Introduction

- transportation forecasting model
- mathematically describes the behaviour of traffic
- people wish to travel on shortest path with least travel time
- goal: find a faster algorithm to solve the shortest path problem between origins and destinations in a transportation network

Traffic assignment

- Traffic Assignment (TA) deals with selection of shortest path for travellers in the network to minimise their travel times
- a non-linear problem, travel times increase dramatically when congestion happens
- an iterative algorithm called Path Equilibration (PE) algorithm is used to solve TA
- PE requires to find millions of shortest paths
- solve shortest path faster to speed up TA
- benefit transportation modelling

Shortest path algorithms

- find path with least distance in network
- search nodes in network in some order until destination is found
- need a data structure called priority queue to keep the searched nodes in sequence so the next node to search can be found easily

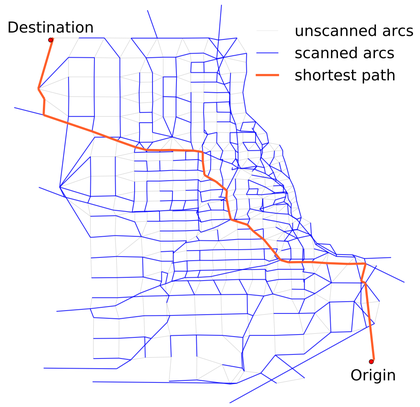
Faster in traffic assignment

- in PE, can avoid shortest path calculations to speed up overall performance
- use shortest path from previous iteration if calculation in current iteration is avoided
- first strategy: avoid the next few iterations if the shortest path of the previous two iterations are identical
- second strategy: randomly avoid the next shortest path calculation in the hope that path of previous and current iteration are identical

Search areas of shortest path algorithms

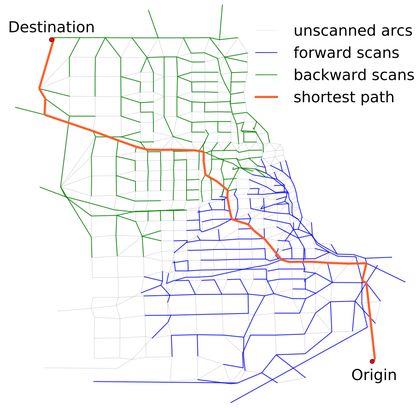
- performance of shortest path algorithms is heavily dependent on the search area
- less searched areas may result faster computational time
- the following figures demonstrate search areas of the implemented shortest path algorithms on a medium sized network with 546 nodes and 2,950 arcs

Dijkstra's algorithm



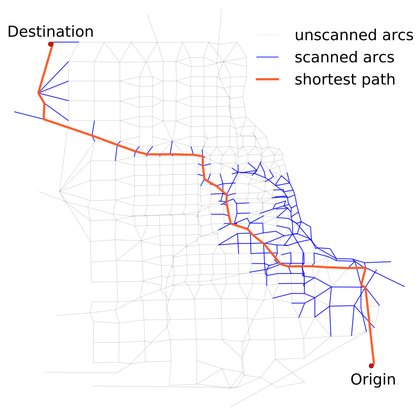
- searches the entire network

Bidirectional Dijkstra's algorithm



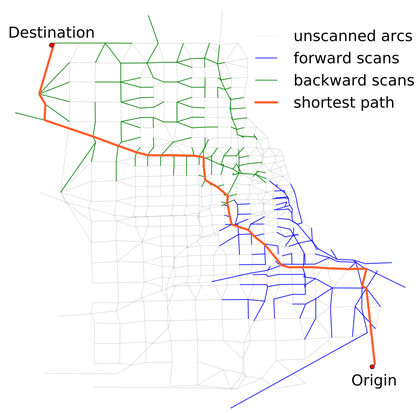
- searches from both ends alternatively

A* Search



- searches along the expected shortest path

Bidirectional A* Search



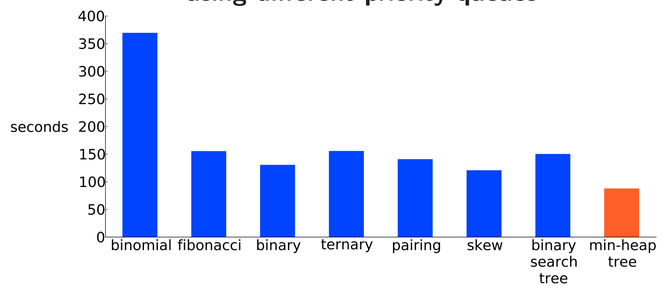
- searches along the expected shortest path from both ends alternatively

- Dijkstra's algorithm searches the most
- bidirectional Dijkstra's algorithm searches slightly less
- A* search searches the least area
- bidirectional A* search searches more than unidirectional A*

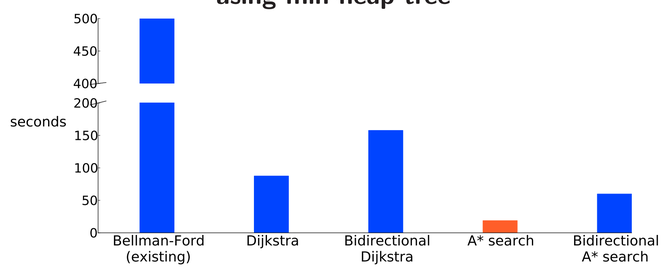
Results on medium sized network

- tested 8 different priority queues, implemented 4 shortest path algorithms and experimented 2 strategies for traffic assignment

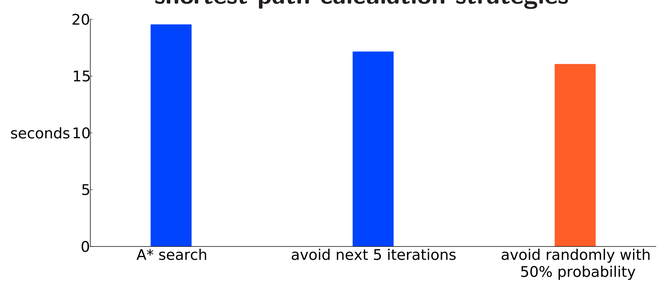
Run times of Dijkstra's algorithm using different priority queues



Run times of shortest path algorithms using min-heap tree



Run times of A* search using avoiding shortest path calculation strategies



Conclusions

- best performance: A* search algorithm using min-heap tree with random avoiding strategy
- overall more than 30 times improvement compared to the existing shortest path algorithm