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 for solving the shortest path problem between origins and a destinations in transportation network

Traffic Assignment

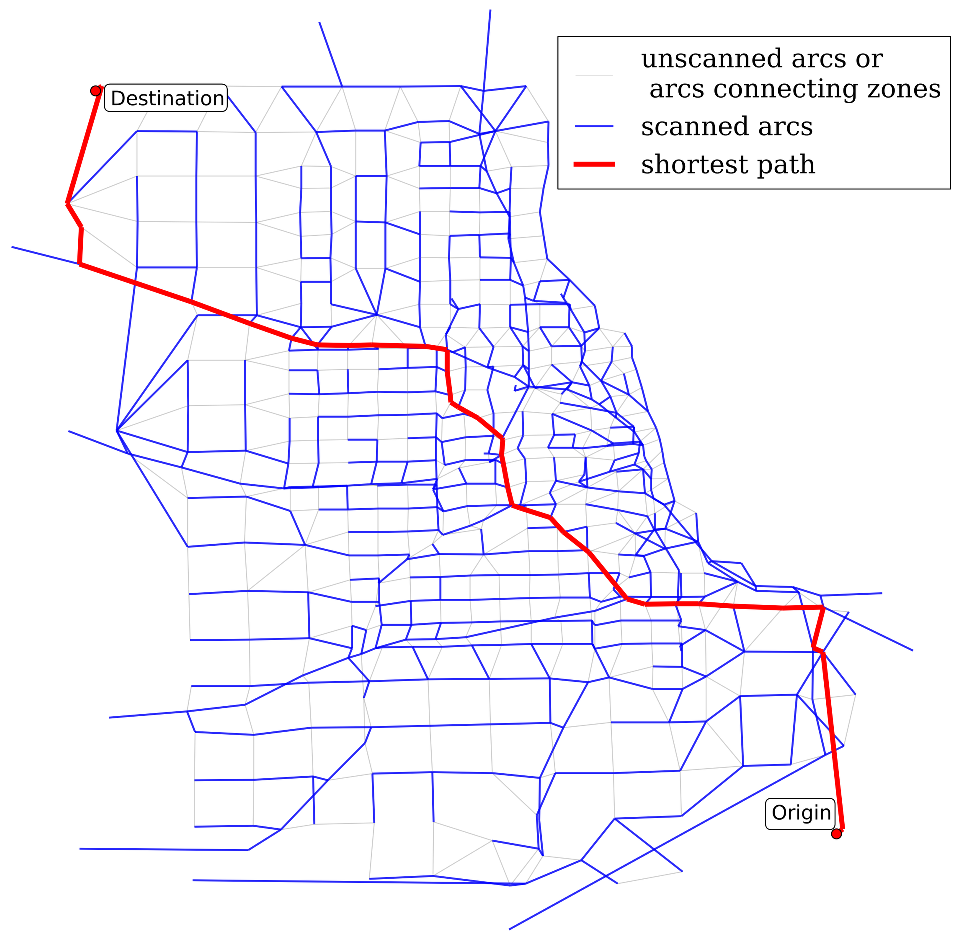
- Traffic Assignment (TA) deals with selection of shortest path for everyone in the network to minimise their travel times
- a non-linear problem, travel times decrease dramatically when congestion happens
- an iterative algorithm called Path Equilibration (PE) algorithm is used to solve TA
- PE requires to find millions of shortest paths
- research of using PE for TA has just begun in recent years due to its huge computation memory requirement
- speed up TA and benefit transportation modelling

Shortest Path Algorithms

- find path with least distance in network
- scan nodes in network in some order until destination is found
- need a data structure called priority queue to keep the scanned nodes in order so we can find the next node to scan easily
- in PE, can avoid shortest path calculations to speed up overall performance
- first strategy: avoid the next few iterations if the shortest path of the previous two iterations are the same
- second strategy: randomly avoid the next shortest path calculation in the hope that path of previous and current iteration are the same

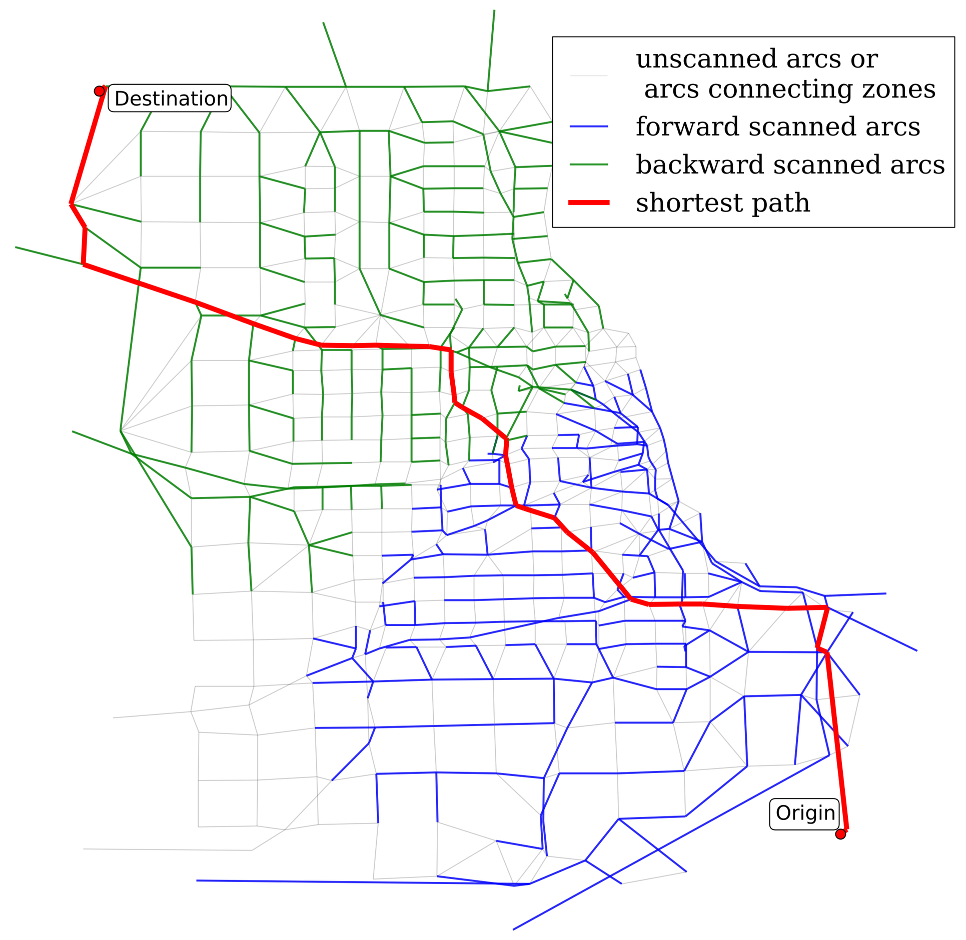
Search Areas of Shortest Path Algorithms

Dijkstra's algorithm



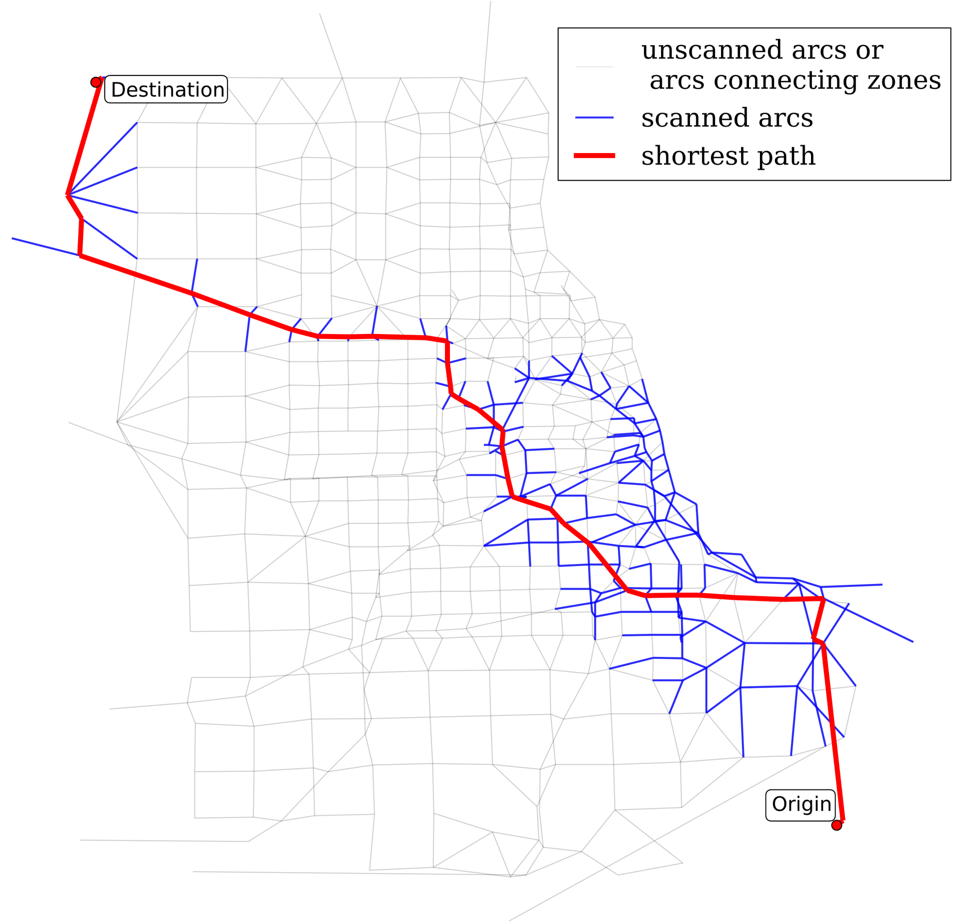
searches the entire network

Bidirectional Dijkstra's algorithm



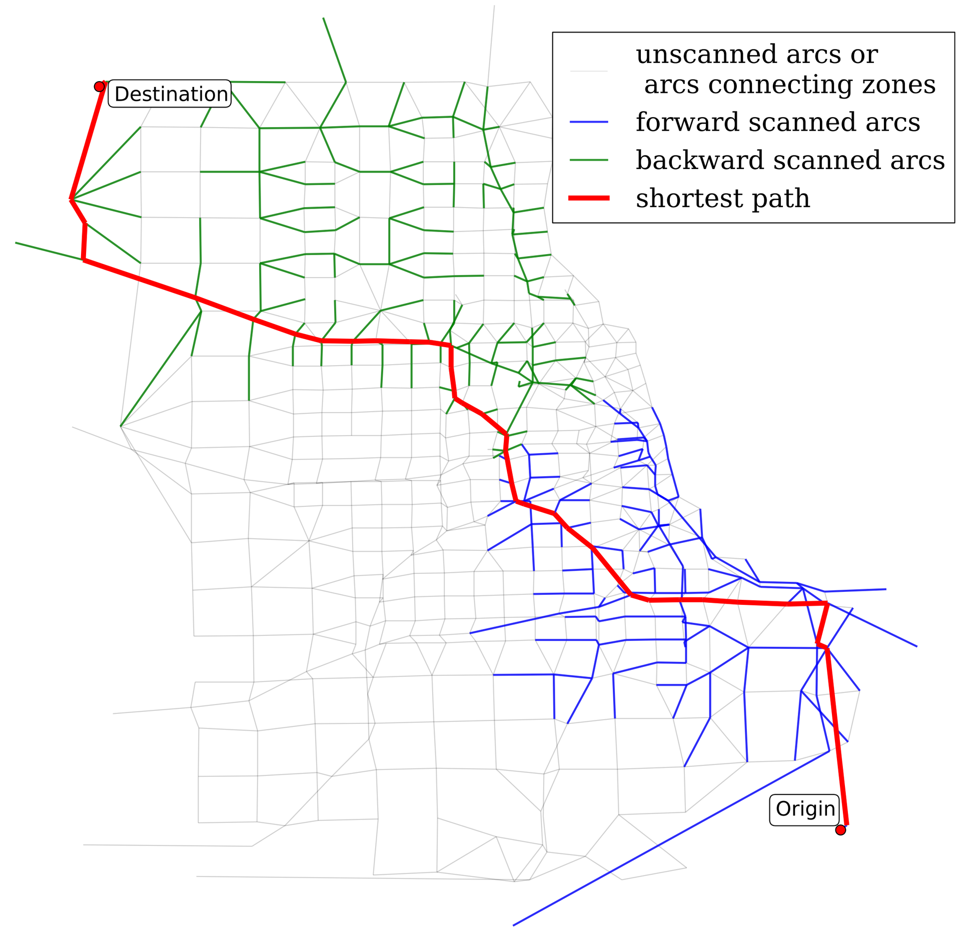
searches from both ends simultaneously

A* Search



searches along the expected shortest path

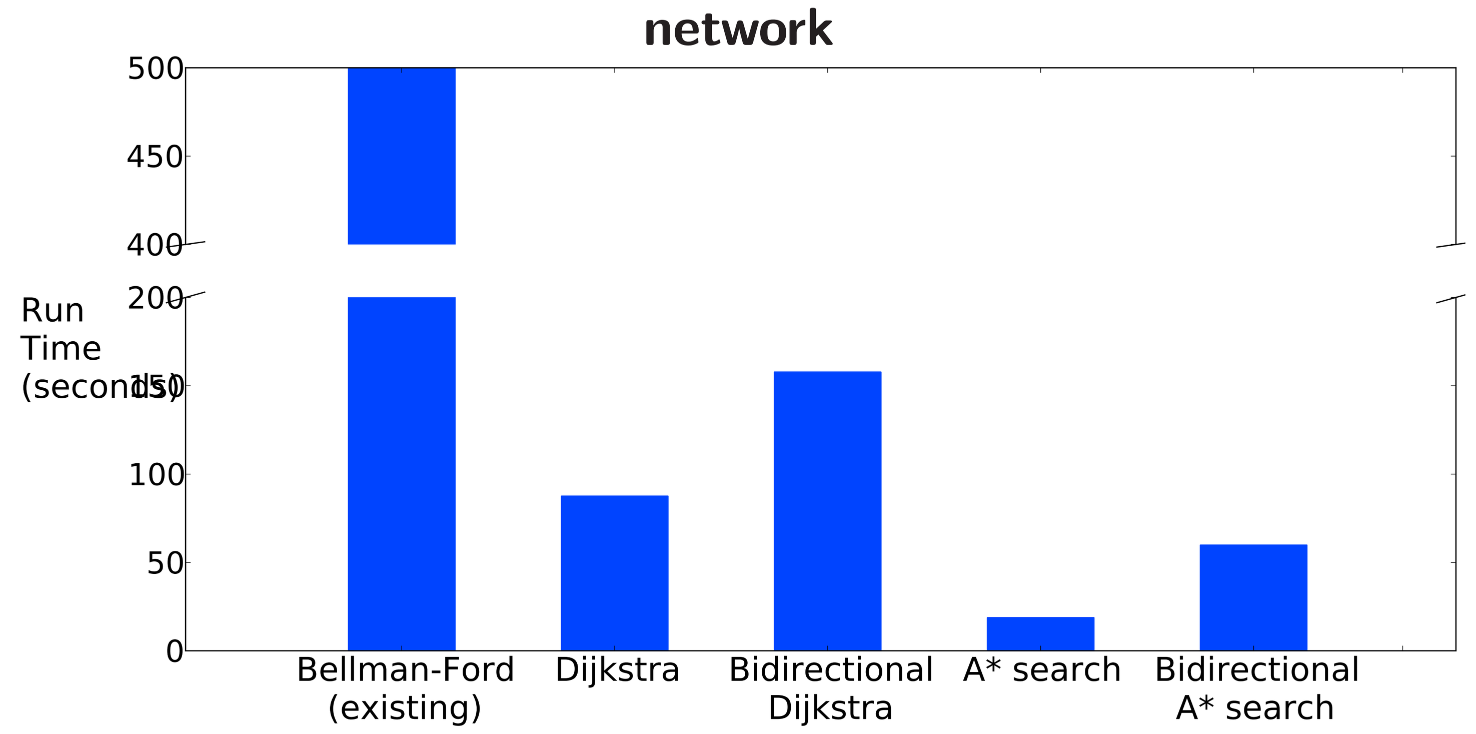
Bidirectional A* Search



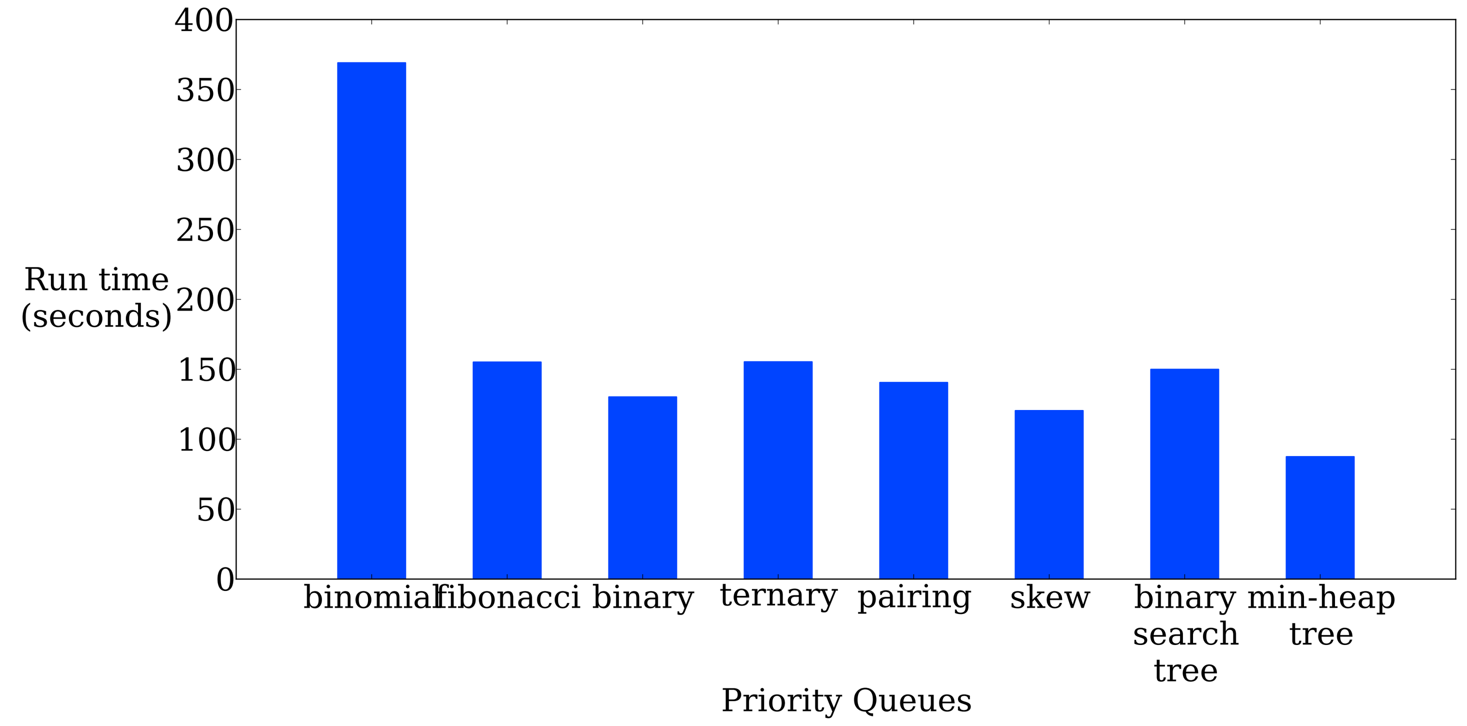
searches along the expected shortest path from both ends simultaneously

Results

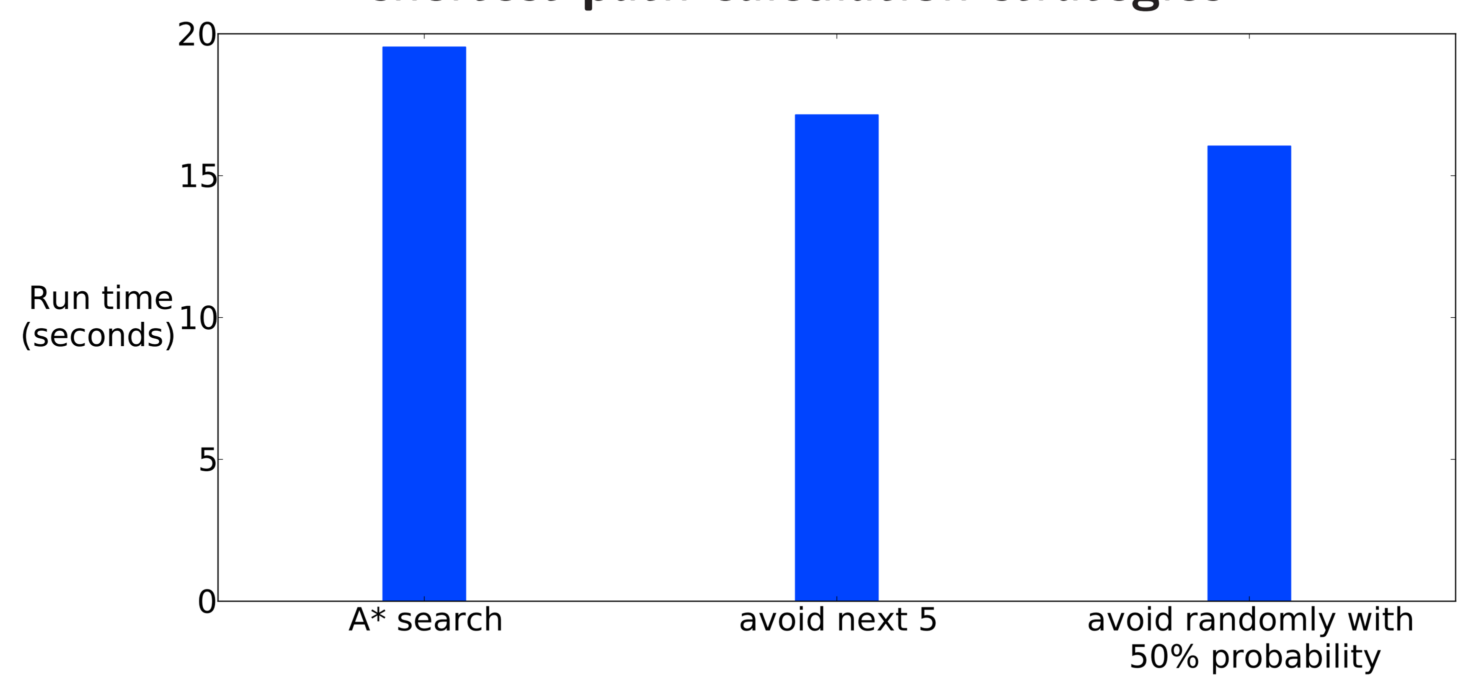
Run times of shortest path algorithms on a test network



Run times of Dijkstra's algorithm run times using different priority queues



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



Conclusions

- best performance : A* search algorithm with random skipping strategy