ABSTRACT

Route optimization, the fundamental component of many intelligent transportation systems, has been a major research area in the field of land transportation. In Singapore, the considerable growing number of vehicle population (with more than 50,000 new registered vehicle in year 2007 and more than 40,000 in year 2008 according to Monthly Digest of Statistics of Singapore, March 2009) has been severely affecting land transportation by increasing the number of accidents, traffic jams, and the fluctuation in traffic condition. This makes the task of providing an effective and efficient route advisory system for dynamic road environment become more acute.

In this paper, we describe our research and development on effective route planning algorithm for dynamic road environment. The research investigates on how to handle and update shortest route when there are any changes in traffic condition. The study also takes into account various issues including accuracy, effectiveness and user’s perspective in handling route adjustment.
1 INTRODUCTION

The Enhanced Route Advisory System (e-RADS) is an enhancement of the existing RADS system developed by the Algorithm Research Group of School of Computing, National University of Singapore which was designed and implemented for private transportation. The current system is able to find the optimal route between a given pair of source and destination location in Singapore road network. The optimizing preference can be to minimize driving distance, time or price. In this research, we would like to the system to support dynamic road environment. The study focuses on the re-computation step which is to update the current optimal route the driver is following when there is a variation on traffic condition. In order to avoid making unnecessary adjustment by re-start the route planner whenever there is traffic update, we have come up with three empirical approaches to handle the problem namely Partial ReCompute, Full ReCompute and ReCompute with Decreased Cost. Each approach is aimed to handle specific types of traffic updates. We also did careful experiment to measure the performances of different approaches.

2 DYNAMIC ROUTE PLANNING ALGORITHM

In this section we will discuss on the dynamic shortest path planning problem and related factors that needs to be considered when designing a solution. The problem can be divided into two steps: the initial shortest path finding and re-computation when there is traffic update.

2.1 Initial Shortest Path Finding

The first computation which is to find an optimal route between two locations is a typical shortest path finding problem. According to our literature survey Bi-directional Dijkstra with Heuristics has been selected due to its outperformance in terms of search time and search space over other algorithms.

2.2 Re-Computation

There are a number of ways to handle the re-computation step, each with its own pros and cons. Some issues that need to be considered when designing a suitable solution include accuracy, efficiency and user’s behavior in route selection and their perception toward route changing.
2.2.1 Accuracy and Efficiency
It is noted that there exists a solution which gives a new shortest path with absolute accuracy simply by starting the route planning from scratch. The result obtained from the re-computation is the optimal least cost path in the updated road network. This approach is simple, accurate but may be inefficient. For example, when the shortest path calculation between two specific locations requires substantial search time and space while the affected area of traffic update is very far away from the route, re-computation is not necessary and thus it is extremely inefficient to re-compute the shortest route.

2.2.2 User’s perception towards change
In addition to accuracy and efficiency, we also take into account user’s perception in designing the solution. More specifically, let consider the following case:

![Figure 1](image)

The user is following a route that travels along an expressway then an accident occurs somewhere ahead. The traffic condition is immediately updated and the system is flexible enough in giving a new faster route to the user as shown in the figure. This new route however travel through a number of minor roads which results in a longer distance while the saving time to is just few minutes from the old route that travel by the accident. It is easy to see that in driver’s point of view changing the whole route just to save few minutes is not necessary.

2.2.3 Re-computation design
Based on the analysis, we designed multiple solutions for solving the updated shortest path problem. There are two main concerns in this process: first is to determine the two locations namely the source and destination between which route are re-calculated and second is to decide whether a re-computation is necessary to be executed.
2.2.4 Determine the locations

Starting point

Taking into account that the current position of the driver at the time traffic update occurs may vary from the original starting location indicated by the user, the very first step of re-computation process is to locate the current position of the driver. In our implementation, it is estimated based on traffic speed and the time the driver has travelled.

Destination point

There are two approaches in choosing destination for shortest path re-computation. The first is simply uses the destination indicated by the driver at the beginning called the final destination. The other is to select an intermediate point in the current shortest path called intermediate destination. While the final destination approach is straightforward, the intermediate destination has been proposed with careful consideration. First, by restarting Bi-directional Dijkstra from current position to an intermediate destination, we can make use of the shortest path found in the first step given that the shortest path from the intermediate destination to final destination is not affected by the update. Second, this reduces the chances of returning the user a new optimal route which involves too much change from the current route while there exists a less variation route with equivalent cost. The following figures demonstrate the situations:

![Initial shortest path from A to D: A=B=C=D Total Cost=52](image1)

![Current location when traffic update occur: B New shortest path from A to D Passing intermediate destination C: A=B=C=D Not passing C: A=E=D Total cost=57](image2)

Figure 2

2.2.5 Type of traffic updates

There are two types of traffic updates that are classified in this study including regular and irregular traffic updates. Regular update includes beginning and ending of peak hour where traffic condition significantly increase or decrease respectively for critical area such as center roads. This type of update also refers to the variation of traffic speed over time which is a reflection of the dynamic of traffic condition. For this kind of updates, traffic speed of any street
in the road network varies slightly and does not significantly affect the optimal cost. The other type of traffic update namely irregular update refers to sudden changes in traffic conditions due to accident or some special events.

2.2.6 Dynamic Route Planning Algorithms

All of the issues discussed above have been taken into consideration to come up with the following approaches to handle the traffic update

**Partial ReCompute:**
Re-compute from current location to **intermediate** when at least one node in **current shortest path** is affected by the update

**Full ReCompute**
Re-compute from current location to **final** destination when at least one node in **current shortest path** is affected by the update

**Full ReCompute - DecreaseCost**
Re-compute from current location to **final** destination when at least one node in **the expanded trees** is decreased by the update

It is noted that Partial ReCompute is the only solution whose re-computation step may return an inaccurate result. The remaining three guarantee that the optimal route will be returned if re-computation is executed. Their only difference is the likelihood to execute a route adjustment.

3 EXPERIMENTS AND RESULTS

The experiment was conducted with a number of experimental runs. Each run has a specific set of input locations (set of source and destination for route finding) and a specific traffic update scenarios. In this experiment, we have tested three proposed empirical approaches with restriction on execution of shortest route re-computation and another approach, Always ReCompute which updates the shortest path by re-calculating the route from scratch whenever there is traffic update. Two important observations have been collected from the experiment.

1. Empirical approaches perform as well as Always ReCompute in updating shortest route. No routes affected by traffic update are excluded from adjusting step due to criteria to trigger a re-calculation in empirical approaches.
2. Partial ReCompute return non-optimal route in test case. It returned a route that has larger cost than those returned by Full ReCompute, Full ReCompute-Decreased Cost and Always ReCompute. This is the consequence of the inaccuracy when reusing part of the initial shortest path, which was mentioned earlier. However, the difference in among the routes however was very subtle. In addition, by reusing part of initial shortest route, Partial ReCompute help the driver to get rid of changing his journey which is required if he follows the optimal updated route provided by other approaches. This result is a support for us to adopt Partial ReCompute even though it does not guarantee the fastest path.

More detailed of the experiments and results are included in the appendix.

4 CONCLUSION

In this project, we have designed alternative solutions to solve the dynamic shortest path problem, done careful experiment to study the performance of each of them and come up with a suitable solution to be implemented into our system. It has been shown that the empirical approaches with restrictions on execution of shortest route adjustment performs well in capturing necessary computation and can help to improve the system efficiency by excluding unnecessary calculation. By taking into account user’s perspective on route changing, the paper also suggests to apply a non-optimal solution which may return a slower route but minimize changes of current route. Lastly, the suitability of each solution varies according to the kinds of traffic updates (i.e. beginning/ending of peak hour, accident, sudden increase in traffic flows,…).

In our research, we have chosen Partial ReCompute and Full ReCompute to handle those updates that increase traffic congestion (i.e. beginning of peak hour or sudden traffic jam) while Full ReCompute – Decreased Cost for updates that decrease the cost (i.e. ending of peak hour or when traffic jam has been cleared). This is to avoid unnecessary re-computation when current route is not affected by increases in traffic congestion and also to avoid missing better route due to strict re-computation criteria when the update decreases traffic congestion. More investigation on traffic update pattern needs to be carried out in developing an efficient and effective route planner for dynamic road environment.
REFERENCES


APPENDIX

1. Experimental Result 1
Initial Shortest Route
Updated Route- Partial ReCompute

Updated Route – Full ReCompute
Updated Routes Comparison

Detailed Computation Result:
Input: 10 pairs of start and destination from north east to south west

<table>
<thead>
<tr>
<th></th>
<th>Partial ReCompute</th>
<th>Full ReCompute/ Full ReCompute- DecreasedCost/ Always ReCompute</th>
</tr>
</thead>
<tbody>
<tr>
<td># of updated links</td>
<td>30000</td>
<td># of updated links : 30000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>#increase: 10770 #decrease: 19230</td>
</tr>
<tr>
<td>Out of 10 Shortest Paths:</td>
<td></td>
<td>Out of 10 Shortest Paths:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>#unchanged: 2 #increase: 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Time: 2254s
Saving time: 8s
NUROP – A Route Planner for Dynamic Road Environment

| #decrease: 2 | #change with same cost time: 0 |

Details of route with difference in shortest path updated with Partial ReCompute and Full ReCompute/ Full ReCompute-DecreasedCost/ Always ReCompute

**Shortest path updated with Partial ReCompute**

File name: Route solution

Algorithm used: 1 to 1 Bidirectional Dijkstra
Number of route nodes: 29
Total time (s): 2256
Total price ($) : 270
Total distance (m): 31690

File name: Route solution - ReCompute Bidijsktra

Algorithm used: 1 to 1 Bidirectional Dijkstra
Number of route nodes: 17
Time left (s): **1492** Travelled Time: 762
Total route time (s): **2254**
Total price ($) : 155
Total distance (m): 20960

Shortest path updated with Full ReCompute/ Full ReCompute-DecreasedCost/ Always ReCompute

File name: Route solution

Algorithm used: 1 to 1 Bidirectional Dijkstra
Number of route nodes: 29
Total time (s): 2256
Total price ($) : 270
Total distance (m): 31690

File name: Route solution - ReCompute Bidijsktra
Algorithm used: 1 to 1 Bidirectional Dijkstra
Number of route nodes: 16
Time left(s): 1484 Travelled Time: 762
Total route time(s): 2246
Total price ($): 145
Total distance (m): 21320

2. Experimental Result 2

<table>
<thead>
<tr>
<th># shortest paths</th>
<th>60</th>
</tr>
</thead>
<tbody>
<tr>
<td># updates</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>Partial ReCompute</td>
</tr>
<tr>
<td># recompute excluded</td>
<td>35</td>
</tr>
<tr>
<td>recompute executed</td>
<td></td>
</tr>
<tr>
<td># increase in cost</td>
<td>22</td>
</tr>
<tr>
<td># decrease in cost</td>
<td>2</td>
</tr>
<tr>
<td># unchanged cost</td>
<td>1</td>
</tr>
</tbody>
</table>
**Detailed of each experimental run**

**RUN 1**
- # of updated links: 18920
- # increase: 18740
- # decrease: 180
- # shortest path: 20

<table>
<thead>
<tr>
<th></th>
<th>Partial ReCompute</th>
<th>Full ReCompute</th>
<th>FullReCompute DecreasedCost</th>
<th>Always ReCompute</th>
</tr>
</thead>
<tbody>
<tr>
<td># not-adjusted</td>
<td>10</td>
<td>10</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td># increase</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td># decrease</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td># unchanged</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>

**RUN 2**
- # of updated links: 18920
- # increase: 18740
- # decrease: 180
- # shortest path: 20

<table>
<thead>
<tr>
<th></th>
<th>Partial ReCompute</th>
<th>Full ReCompute</th>
<th>FullReCompute DecreasedCost</th>
<th>Always ReCompute</th>
</tr>
</thead>
<tbody>
<tr>
<td># not-adjusted</td>
<td>13</td>
<td>10</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td># increase</td>
<td>7</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td># decrease</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td># unchanged</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>9</td>
</tr>
</tbody>
</table>

**RUN 3**
- # of updated links: 18920
- # increase: 18740
- # decrease: 180
- # shortest path: 10

<table>
<thead>
<tr>
<th></th>
<th>Partial ReCompute</th>
<th>Full ReCompute</th>
<th>FullReCompute DecreasedCost</th>
<th>Always ReCompute</th>
</tr>
</thead>
<tbody>
<tr>
<td># not-adjusted</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td># increase</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td># decrease</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td># unchanged</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
</tbody>
</table>
### RUN 4

# of updated links: 30000  
# increase: 10770  # decrease: 19230

<table>
<thead>
<tr>
<th># shortest path</th>
<th>Partial ReCompute</th>
<th>Full ReCompute</th>
<th>FullReCompute DecreasedCost</th>
<th>Always ReCompute</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># not-adjusted</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># increase</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td># decrease</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td># unchanged</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>