Chapter 8

Conclusion and Further work

There are many solutions that have been developed to address different problem areas in carpooling systems. However, most of them have some advantages as well as disadvantages. This paper reviews and analyzes several researches and projects undertaken with regard to the route matching and passenger matching approaches. The purpose of this research is to provide an enhanced route searching and route suggesting method, which is one of the most important feature of the carpooling systems. The enhanced route searching mechanism ensures that the results are most ideal and similar to the preferred routes of the passengers. The alternative path suggesting mechanism is implemented to improve the driver's interaction with carpooling systems, which will advance the passenger matching from driver's perspective which is a new area to be considered.

As further work, applying an ideal geo-coordinate clustering method and a method to reduce the number of geo-coordinates in each route, needs to be implemented to improve the searching performance. An online payment mechanism through this carpooling application can be introduced to handle and track all the payments for rides. Also, a mobile application can be implemented to access these carpooling services through mobile devices which will improve the user satisfaction.

References

[1] S. A. Shaheen and A. P. Cohen, "Carsharing and Personal Vehicle Services: Worldwide Market Developments and Emerging Trends," *Int. J. Sustain. Transp.*, vol. 7, no. 1, pp. 5–34, Jan. 2013.

[2] M. M. Galizzi, "The economics of Car-Pooling: A survey for Europe," in *Paper for the workshop: Highways-Cost and Regulation in Europe. Universita degli Studi di Bergamo, Bergamo, Italy*, 2004.

[3] B. Srivastava, "Making Car Pooling Work–Myths and Where to Start," in *Proc. 19 ITS World Congress Semantic Cities Workshop, Vienna, Austria*, 2012.

[4] S. R. I. Sweta, M. Mounika, P. Agrawal, and G. B. Pallavi, "A Survey to Justify the Need for Carpooling."

[5] K. K. Dewan and I. Ahmad, "Carpooling: A Step To Reduce Congestion," *Eng. Lett.*, vol. 14, no. 1, 2007.

[6] D. Graziotin, "An Analysis of issues against the adoption of Dynamic Carpooling," *ArXiv Prepr. ArXiv13060361*, 2013.

[7] P. Brinckerhoff, "Greenhouse gas (ghg) and energy mitigation for the transportation sector," 2009.

[8] Y.-T. Chen and C.-H. Hsu, "Improve the Carpooling Applications with Using a Social Community Based Travel Cost Reduction Mechanism," *Int. J. Soc. Sci. Humanity*, pp. 87–91, 2013.

[9] D. Zhang, Y. Li, F. Zhang, M. Lu, Y. Liu, and T. He, "coRide: carpool service with a win-win fare model for large-scale taxicab networks," 2013, pp. 1–14.

[10] R. Manzini and A. Pareschi, "A Decision-Support System for the Car Pooling Problem," *J. Transp. Technol.*, vol. 02, no. 02, pp. 85–101, 2012.

[11] S. Di Martino, R. Galiero, C. Giorio, F. Ferrucci, and F. Sarro, *DMS 2011: proceedings : the 17th International Conference on Distributed Multimedia Systems : technical program, August 18-20, 2011, Convitto della Calza, Florence, Italy*. Skokie, IL: Knowledge Systems Institute Graduate School, 2011.

[12] J. Xia, K. M. Curtin, W. Li, and Y. Zhao, "A New Model for a Carpool Matching Service," *PloS One*, vol. 10, no. 6, p. e0129257, 2015.

[13] C. Mulders and B. Lambeau, "Carpooling, a vehicle routing approach."

[14] "PHP: Hypertext Preprocessor." [Online]. Available: http://php.net/. [Accessed: 14-Mar-2016].

[15] "MySQL." [Online]. Available: https://www.mysql.com/. [Accessed: 14-Mar-2016].

[16] "Welcome to The Apache Software Foundation!" [Online]. Available: http://www.apache.org/. [Accessed: 14-Mar-2016].

[17] "WampServer," *WampServer*. [Online]. Available: http://www.wampserver.com/en/. [Accessed: 14-Mar-2016].

[18] "Google Maps JavaScript API," *Google Developers*. [Online]. Available: https://developers.google.com/maps/documentation/javascript/. [Accessed: 14-Mar-2016].

[19] "RouteBoxer Documentation: Examples." [Online]. Available: http://google-maps-utility-library-v3.googlecode.com/svn/trunk/routeboxer/docs/examples.html. [Accessed: 14-Mar-2016].

[20] "MySQL :: MySQL 5.7 Reference Manual :: 12.15.1 Spatial Function Reference." [Online]. Available: https://dev.mysql.com/doc/refman/5.7/en/spatial-function-reference.html. [Accessed: 14-Mar-2016].

[21] "Haversine formula," *Wikipedia, the free encyclopedia*. 14-Mar-2016.

[22] "Bearing (navigation)," *Wikipedia, the free encyclopedia*. 18-Feb-2016.

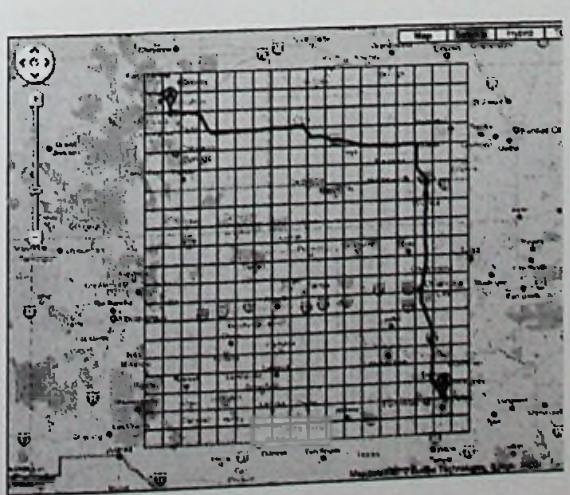
Appendix A – SQL to filter routes which are going through given two points

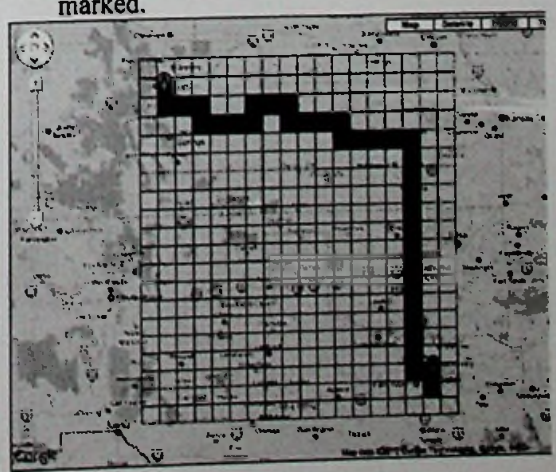
```
SELECT T1.route_id
FROM (
    SELECT distinct route_id, ( 6371 * acos( cos( radians(6.8412738) ) *
    cos( radians( X(point) ) ) * cos( radians( Y(point) ) - radians(79.8640329) ) +
    sin( radians(6.8412738) ) * sin( radians( X(point) ) ) ) ) AS distance
    FROM sp_lift_offer_points
    GROUP BY(route_id)
    HAVING distance < 20
    ORDER BY distance
) AS T1
JOIN (
    SELECT distinct route_id, ( 6371 * acos( cos( radians(6.8412738) ) *
    cos( radians( X(point) ) ) * cos( radians( Y(point) ) - radians(79.8640329) ) +
    sin( radians(6.8412738) ) * sin( radians( X(point) ) ) ) ) AS distance
    FROM sp_lift_offer_points
    GROUP BY(route_id)
    HAVING distance < 10
    ORDER BY distance
) AS T2
ON T1.route_id = T2.route_id
LIMIT 0 , 20
```

Appendix B – Steps performed by route boxer utility library

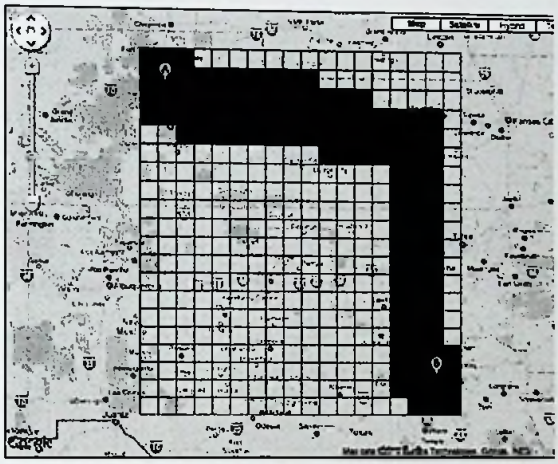
1. A grid is overlaid on the route with square cells spaced at the specified distance. The grid is centered on the bounding box of the route polyline, and extends one cell further out in each direction than is necessary to cover the entire route.

2. Every cell in the grid that the route intersects with is identified. To do this the vertices on the route polyline are traversed, and the cell containing each vertex is marked. If a cell that is marked does not share an edge with the cell for the previous vertex, the intermediate cells are also marked.

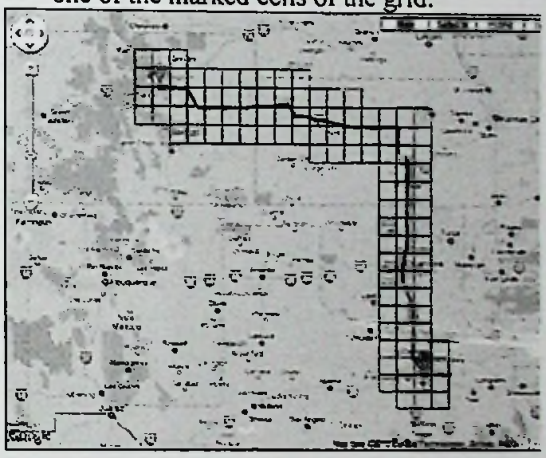




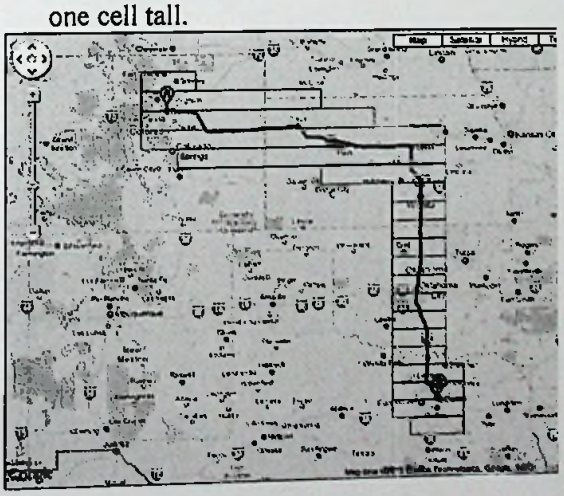
3. When a cell is marked, each of the 8 cells surrounding it are also marked.



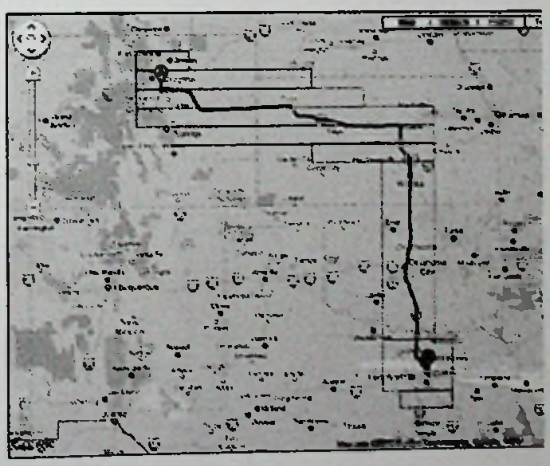
4. When all of the route vertices have been traversed, any point within the specified distance of the route is guaranteed to be in one of the marked cells of the grid.



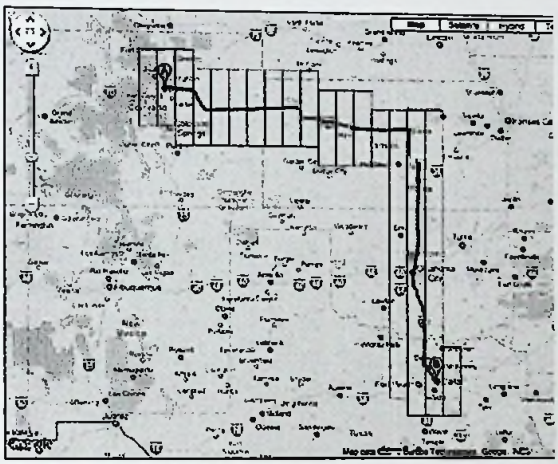
5. The marked cells are then merged into a set of non-overlapping rectangular boxes. Two different approaches are taken to this. The first approach merges cells that adjoin horizontally into a set of wide boxes, each one cell tall.



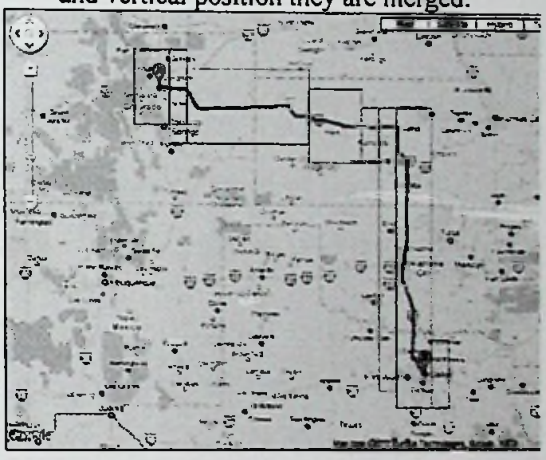
6. Each box is then compared to the boxes on the row below, and if there is a box of the same width and horizontal position they are merged.



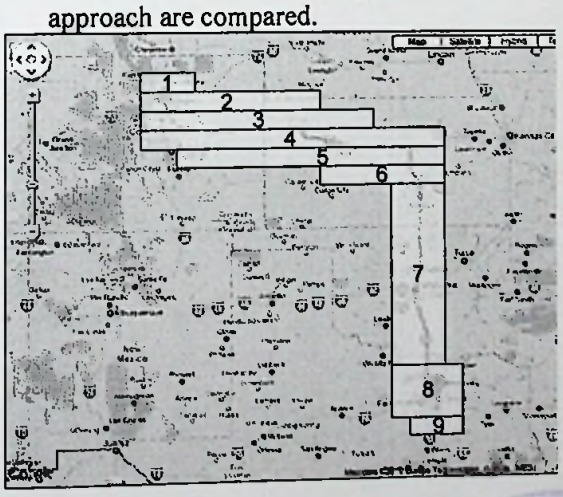
7. The second approach follows the same technique, but first merges cells vertically into a set of tall boxes, each one cell wide.



8. Each of these boxes are then compared with the boxes in the column to the left, and if there is a box with the same height and vertical position they are merged.



9. When the two approaches are complete the number of boxes that resulted from each approach are compared.



10. The set of boxes that is smallest in number are returned to the application.