Intelligent Car Park Routeing for Road Traffic

Victoria J. Hodge[1], *Mike Smith*[2], *Jim Austin*[3]

Abstract
The twin problems of congestion and inner-city parking limitations affect many cities. One solution is to promote the use of Park and Ride sites. However, for effective use of the sites, drivers need to know where the sites are and which is the "best" site to use. This work introduces a methodology to pinpoint and guide drivers to the best Park and Ride site from their current location. While drivers may be able to obtain traffic, car park location and free space data individually, the information is not usually coordinated. By fusing up-to-date details of traffic jams, roadworks and accidents coupled with free parking spaces and combining this with a novel route weighting methodology, we are able to ensure that intelligent information is displayed to guide drivers. The method uses optimised data structures and proprietary scoring measures to ensure it is fast and accurate. The method provides a simple and low cost solution through the use of existing technologies to display information to drivers.

1. Introduction

In the UK, the volume of road traffic is increasing faster than the capacity of the roads which carry that traffic. This growth will continue unless measures are introduced to reduce traffic, optimise vehicle use and thus, reduce congestion. Another problem in many large cities is the lack of available space to provide sufficient parking spaces. Drivers circulate between car parks or queue at the entrance to particular car parks in search of a parking place.

Many cities such as York, UK are attempting to combat the twin problems of traffic congestion and parking limitations by constructing Park and Ride sites at the edges of cities. Park and Ride sites allow drivers to park their vehicles in the car park for as long as necessary and then to use a public transport system to travel into the city. To maximise usage and promote Park and Ride, a cost-effective system is needed to deliver real-time, accurate and useful information to users.

[1] vicky@cs.york.ac.uk, Dept of Computer Science, University of York, York, YO10 5DD, UK.
[2] mjs7@york.ac.uk, Dept of Mathematics, University of York, York, YO10 5DD, UK.
[3] austin@cs.york.ac.uk, Dept of Computer Science, University of York, York, YO10 5DD, UK.

A system is needed that uses suitable criteria to select and display the "*best*" Park and Ride site for drivers to choose at key locations as drivers approach the city using existing infrastructure such as VMSs so it is cost effective. Shaheen et al. (2004) note that "*awareness and understanding of Parking Guidance Information signs can be relatively high but, to be effective, messages must display accurate information that meets travellers' needs*". One of the aims of the FREEFLOW project is to develop such a routeing and display system.

There are many technologies available that aim to provide routes for road users. The bases for such technology are routeing algorithms. Road networks are directly related to graph theory and route finding is based on graph theory rules Hofmann-Wellenhof (2003). There are various algorithms used in the traffic routeing literature for calculating the shortest path (or minimum cost path) through such a graph. No "best" algorithm exists as transportation problems vary widely so an algorithm has to be chosen that best meets the requirements of the particular problem.

The seminal routeing algorithm is Dijkstra (1959) algorithm which is guaranteed to find the best route through a directed graph of vertices and weighted edges. The measure used to calculate the edge costs varies from application to application. Garofalakis et al. (2007) use the time to travel the road section calculated from both the length of the road section and the average vehicle speed on that road section as the edge costs. Eglese et al. (2005) use historical records to calculate the expected travel times for each road section. Zhan and Noon (1998) noted that Dijkstra's algorithm has advantages over similar algorithms, as the search may be terminated once the shortest path to the set of destination vertices is obtained. Thus, Dijkstra's algorithm often produces considerable efficiency savings in comparison to similar algorithms and will allow us to scale to larger road networks.

Other routeing algorithms include the A* algorithm which is another well-known greedy algorithm and is used by Chabini and Lan (2002). Fawcett and Robinson (2000) adapt and optimise Lee's algorithm for traffic routeing. Boehlé et al. (2008) propose the City Based Parking and Routeing System (CBPRS). CBPRS allows users to reserve parking places and provides a dynamic, ant-based routeing algorithm to determine the best route from the vehicle's current location to the chosen parking place. However, the system requires a hardware infrastructure to send and receive signals.

In this paper, we introduce a methodology to guide road users to their best choice of Park and Ride site to reduce the number of vehicles entering city centres, to reduce unnecessary road mileage and to minimise driver

frustration. The main contributions of this paper are: we have extended Dijkstra (1959) algorithm to Park and Ride site recommendation by using suitable adaptations of the edge and path weights incorporating traffic data from multiple traffic data feeds with road statistics such as road section lengths and the number of free spaces in the park and Ride sites. We use an optimised data structure within Dijkstra's algorithm and cease processing when the set of destination vertices have all been reached. We use existing VMSs wherever possible to display the Park and Ride routeing information and therefore, ensure that the methodology is cost-effective.

2. Park and Ride Recommendation using Dijkstra's Algorithm

The methodology has been implemented for York, UK and the road network and Park and Ride sites are shown in Fig. 1a. The road network shown comprises 5 Park and Ride sites, 48 junctions and 81 road sections.

The Dijkstra (1959) algorithm finds the path with lowest cost between a start vertex and a set of destination vertices in the graph. We have subdivided the road network into road sections. Each road section maps to a graph edge and each edge has an associated length which represents the length of the corresponding road section. The road network is, therefore,

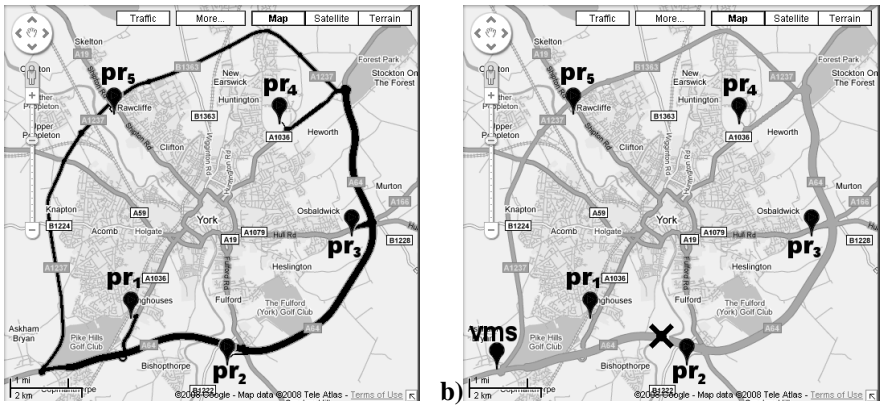

Fig. 1a. *Shows the road network and Park and Ride sites. The entire road network is shown but routeing is limited to the road sections shaded black. This limitation is to prevent vehicles entering the urban traffic zone wherever possible.*

Fig. 1b. *shows the position, marked with a cross (near pr2), of the a VMS and road incidents cited in scenarios 2-4 in section 0. The map is taken from Google Maps (2008).*

a directed graph: $G \equiv (V, E)$, where the finite set V represents the set of vertices (road junctions) in the graph and $E \subseteq V \times V$ is the set of directed edges (road sections) in the graph. The graph and underlying road network are both directed so $(v_i, v_j) \in E$ if and only if there is a road link that permits traffic to travel from junction v_i to junction v_j in the prescribed direction. Both Fawcett &Robinson (2000) and Garofalakis et al. (2007) assume that all road sections are straight. We have used a combination of Google Maps (2008) and Microsoft Multimap (2008) to obtain actual road section lengths. This means that the length of the road section from v_i to v_j may differ from the length of the road section from v_j to v_i. The current speeds of each road section are stored in the corresponding edges in the graph along with the last update time for each data source, and the length of each road section.

The speeds of vehicles on a particular stretch of road vary throughout the day and thus introduce a dynamic element to our methodology. We fuse two traffic data feeds downloaded at 15 minute intervals to allow us to represent the current traffic situation. The Highways Agency (2008) publishes real-time average traffic speeds in km/h for labelled road sections. We cross-reference road sections to those in our graph and thus incorporate the traffic speed updates into our route calculations. The BBC TPEG Traffic (2008) data provides details of incidents with each incident ascribed a severity factor between 1 and 5 (where 5 is severe and 1 is slight) which forms our speed penalty factor. We revise the speed for each graph edge using the following priorities where current means within the last 15 minutes (last download). If there is a current average speed available from the Highways Agency data then we use that for the road section as it is the most accurate. If there is no Highways Agency data available then we use the BBC update by dividing the speed limit by the severity factor.

Dijkstra's algorithm is a cost minimisation algorithm where each edge in the underlying graph has a **cost**. In our methodology, the cost is an estimate of the time to traverse the road section. The cost $C(v_i,v_j)$ assigned to each road section is calculated as ***length of road section (in km) / latest speed (in km/h)***. Latest speed is the most recently stored speed update for the particular road section.

2.1. Route Scores

In calculating the best Park and Ride site to recommend, we also need to consider the number of free spaces available. There is no point directing vehicles to Park and Ride sites with few available spaces as, by the time the

driver has driven from the VMS display to the Park and Ride site, these few spaces may have been occupied. Therefore, the system downloads the number of free spaces in each park and Ride site at 1 minute intervals. We may then include a free space penalty factor in the calculation of the best Park and Ride site to recommend. For a Park and Ride site, the cumulative route score $CC(v_k)$ from the VMS to the Park and Ride incorporates the *free space penalty factor, Prscore,* and is given by $\boldsymbol{PRscore(v_k) * Cumulative\ Cost}$. The *PRscores* are listed in Table 1.

Table 1. *Table listing the free space penalty factors for Park and Ride sites.*

Number of free spaces	*<=25*	*26-50*	*51-75*	*76-100*	*>100*
PRScore	Infinity	20	10	5	1

2.2. Implementation

Routeing algorithms are frequently speeded-up using efficient data structures (such as buckets Garofalakis et al. (2007) or road hierarchies Eglese et al. (2005) for large road networks) The run-time for the standard algorithm is $O(|V|^2)$ where $|V|$ represents the number of vertices. We use a Java Priority Queue data structure which provides an asymptotic run-time of $O((|V|+|E|)log(|V|))$, where $|E|$ is the number of edges and $|V|$ is the number of vertices in the graph. We have only partial connectivity and thus, $|E| \cong O(|V|)$ and the time complexity reduces to $O(|V|\ log\ |V|)$. The algorithm runs with each **VMS** as a start node to update the best Park and Ride information displayed on that VMS. It stores a set of vertices U for which shortest paths have not been found, and stores $CC(v_i)$: the shortest known path from the *start vertex* to v_i . Initially, $U=V$ and all $CC(v_i)=\infty$. At each iteration, the vertex v_j in U with the smallest $CC(v_j)$ value is removed from U. For each neighbour (v_m) of v_j found in U, the algorithm calculates if $((CC(v_j) + C(v_j,v_m)) < CC(v_m))$, i.e., whether a path through v_m would have a lower score than the currently best-known path. Once all Park and Ride site vertices have been removed from U, we stop processing to minimise the execution time. The Park and Ride site with the lowest route score ($CC(v_k)$) calculated by the algorithm will be displayed on the VMS along with simple route information. Examples from York, UK are "*Use Fulford P&R: A64 then A19 Fulford Road*" or "*Use Rawcliffe P&R: A1237 to Shipton Road*". In our previous paper Hodge et al. (2009), we demonstrated that the algorithm was able to find the best Park and Ride site in York in 0.016 seconds.

The system downloads traffic updates at 15 minute intervals but downloads the free space data at 1 minute intervals. We update the display for each VMS by calculating the best Park and Ride to use from that location at 1 minute intervals to accommodate the most frequent updates. We note that we do not run the routeing calculations until data downloads are completed and we do not run the calculations if the data values have not changed (speeds, flows, occupancies, free spaces etc.) since the last update to prevent unnecessary processing. Also, the update frequencies may be easily varied according to data availability.

3. Example Route Calculation and Routeing Scenarios.

In the following we demonstrate route calculating in our methodology using an example **VMS** as the *start vertex* and finding the best Park and Ride from the five Park and Ride sites shown in Fig. 1b (pr_1, pr_2, pr_3, pr_4 *and* pr_5) using some example free space and traffic scenarios. The demonstration highlights how our methodology can dynamically alter the message displayed on the VMS to recommend the optimal Park and Ride site and allow drivers to avoid, full car parks, accidents and slow traffic.

In the first example, **example VMS**, we calculate the route from a **VMS** to the nearest (best) Park and Ride site under normal circumstances. In **scenario 1**, pr_1, the previous best Park and Ride from example VMS, has filled up and there are only 10 free spaces available. In **scenario 2**, pr_1 still has 10 free spaces and now there is an accident on the dual carriageway at the location shown in Fig. 1b with accident data received from the BBC TPEG data where the accident is severity 3. In **scenario 3**, the traffic data has been received from the Highways Agency rather than the BBC so pr_1 still has 10 free spaces and now there is slow traffic on the dual carriageway due to another accident at the location shown in Fig. 1b with traffic speed for the road section of 32 km/h. The Park and Ride scores calculated by our methodology for the example routeing and four scenarios respectively are given in table 2.

Table 2. *Table listing the routeing scores for the Park and Ride sites under various traffic scenarios.*

Park & Ride	Pr_1	Pr_2	Pr_3	Pr_4	Pr_5
Example VMS	**2.84**	5.29	6.63	10.1	8.89
Scenario 1	Infinity	**5.29**	6.63	10.1	8.89
Scenario 2	Infinity	**8.27**	9.61	13.08	8.89
Scenario 3	Infinity	9.05	10.39	13.87	**8.89**

From table 2, the scores vary and thus the recommended Park and Ride site varies according to the different traffic scenarios. A full Park and Ride site or a severe accident will reroute traffic. **Scenario 1**, reroutes to pr_2 from pr_1 as pr_1 is full. As pr_1 remains full for the remaining scenarios then it is not recommended again. **Scenario 2** also routes to pr_2 despite the accident en-route but the difference between the score for pr_2 and pr_5 is much closer than it was in **scenario 1** which had no accident. **Scenario 3** reroutes to pr_5 from pr_2 due to the more severe accident and associated congestion which just increases the score for pr_2 enough to cause re-routeing to pr_5.

4. Conclusion

The methodology outlined in this paper provides a simple and low cost solution to guiding drivers to the best Park and Ride site. Our algorithm updates the information displayed on the message displays at regular intervals using the most up-to-date traffic information. Our system currently fuses four data sources: three traffic data feeds and a parking space counter feed. Our proposed methodology is adaptable and adding new data feeds into the methodology should be relatively simple. The parking guidance system of Boehle et al. (2008) requires an infrastructure of lamppost transmitters and in-vehicle hardware to operate. Our system utilises data feeds, hardware and systems that are readily available in many cities thus minimising cost.

The system will be implemented in the FREEFLOW project to test whether the methodology introduced in this paper provides information that is useful to drivers and whether the correct Park and Ride are recommended. The methodology introduces two proprietary score measures for calculating the best route. We will verify that these score measures are producing the correct guidance.

The ultimate goal for the system is to interact with mobile handheld devices to guide users to the best car park. The system will route drivers from their current location ascertained from the GPS coordinates of their handheld device and guide them to their best car parking option.

5. Acknowledgement

This research was undertaken by University of York as part of the FITS FREELOW Project: a consortium of industrial companies, academia and local authorities. FREELOW is funded by Technology Strategy Board, DfT, EPSRC and the partners.

6. References

BBC TPEG Data (2008). BBC Travel News TPEGML Feeds (for North Yorkshire) http://www.bbc.co.uk/travelnews/tpeg/en/local/rtm/york_tpeg.xml (accessed 30/10/2008).

Boehle, J.L., et al., (2008). CBPRS: A City Based Parking and Routing System. ERIM Report Series Reference No. ERS-2008-029-LIS

Chabini, I., and Lan, S., (2002). Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks. IEEE Trans. on Intelligent Transportation Systems 3(1), 60–74.

Dijkstra, E.W., (1959). A note on two problems in connexion with graphs. In Numerische Mathematik 1(1), 269–271.

Eglese R.W., et al., (2005). A road timetable to aid vehicle routing and scheduling, Computers and Operations Research 33(12), 3508-3519, Elsevier.

Fawcett, J., and Robinson, P., (2000). Adaptive Routing for Road Traffic, IEEE Computer Graphics and Applications 20(3), 46-53.

Garofalakis, J., et al., (2007). Vehicle Routing and Road Traffic Simulation: A Smart Navigation System, 11th Panhellenic Conference on Informatics (PCI 2007), May 18-20, Patras, Greece.

Google Maps, (2008). Google Maps available at http://maps.google.com (last accessed 30/06/09).

Highways Agency, (2008) Transport Informastion Highway (TIH) web site for DATEX-II traffic updates http://www.tih.org.uk/ (last accessed 30/10/2008)

Hodge, V.J., et al. (2009). Intelligent Car Park Routeing and Recommendation for Road Traffic. Submitted to, Elsevier Transportation Research C – Emerging Technologies.

Hofmann-Wellenhof, B., et al., (2003). Navigation: Principles of Positioning and Guidance, Springer, ISBN 3211008284, 9783211008287

Microsoft Multimap, (2008). Microsoft Multimap available at http://www.multimap.com/ (last accessed 30/06/09).

Shaheen, S.A., et al., (2004) Applying Integrated ITS Technologies to Parking Management Systems: A Transit-Based Case Study in the San Francisco Bay Area. Institute of Transportation Studies, University of California, Davis, Research Report UCD-ITS-RR-04-18

Zhan, F.B.. and Noon, C.E.. (1998). Shortest Path Algorithms: An Evaluation Using Real Road Networks. Transportation Science 32(1), February 1998