Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 47 (2020) 401–408

Transportation Research Procedia

www.elsevier.com/locate/procedia

22nd EURO Working Group on Transportation Meeting, EWGT 2019, 18-20 September 2019, Barcelona, Spain

Fast estimation of point-to-point travel times for real-time vehicle routing

Guido Gentile[a,b], Lorenzo Meschini[b], Daniele Tiddi[b]*, Agostino Fiorani[b], Alessandro Attanasi[b]

[a]*Sapienza Università di Roma - DICEA, Via Eudossiana 18, Rome, Italy*
[b]*PTV Group - SISTeMA, Via Bonghi 11b, Roma, Italy*

Abstract

To provide the optimal allocation of requests to the available fleet vehicles, routing algorithms typically assume the availability of complete and correct information about point-to-point travel times. Actually, in real applications non-recurrent events and traffic conditions make the estimation and the prediction of such travel times a difficult task, further complicated in real-time applications by the dynamicity of the information and the number of needed estimates.
In this paper we present a complete methodology to achieve a computation of point-to-point travel times on a large network which proves to be both extremely fast and consistent with dynamically updated traffic information.

Keywords: fleet management; dispatching; dynamic skim matrix; real-time.

1. Introduction

Shared mobility and fleet management make nowadays the routing of a set of vehicles to satisfy incoming service demand a very common and relevant problem. Several algorithms are found in the literature to provide the optimal allocation of requests to vehicles, under different assumptions, see for example Jung (2012).

In real-time applications, there is a strong need of reliable travel times, to avoid compromising the optimality and, more important, the feasibility of vehicle routing solutions. On one side, strict temporal constraints often apply to transportation and delivery services of people and goods. Their violation may not only affect customer satisfaction and contract terms, jeopardizing the overall service quality, but may get the solution factually unfeasible. On the other

* Corresponding author. Tel.: +39-06-99344415; fax: +39-06-99334872.
E-mail address: daniele.tiddi@ptvgroup.com

10.1016/j.trpro.2020.03.115

side, recurrent congestion and non-recurrent events make the estimation and the prediction of travel times a difficult task, from the modelling as well as from the computational point of view.

The vehicle routing problem has a combinatorial nature, as it is related to the set partitioning and ordering of the requests. Thus, the algorithms require to evaluate many tentative solutions, each one demanding for travel times between different origin-destination couples. Moreover, in real-time applications origins and destinations may emerge at any point of the map rather than at specific nodes of the network and then discourage approaches which assume the positions to be already known in advance.

1.1. The travel time information

The availability of correct point-to-point travel times underlies any approach, and these data are often declassified by the operation research community as a commodity. Typically, the three main approaches below can be identified.

First, using a very rough approximation such as the Euclidean distance, see Agarwal *et al.* (2004). This approach has the advantage of its simplicity and at the same time its versatility, as it can be applied to any couple of points in the space with light computational burden. Of course, it underestimates the travel time and additionally neglects any relationship between the two points and the underlying network items constituting the path between them, then not allowing to infer any information from the items to the two points travel time (e.g. a link congestion would increase the travel time of the path, if using it).

Second, assuming to know the full matrix of node-to-node costs, like Miller *et al.* (1960) and Brandão (2011) do. This approach is again very simple, despite it involves memory consumption to store the information amongst all the needed couples of points. This approach could be further divided into having the information amongst all the existing nodes or amongst only some problem relevant points (e.g. the depot and delivery points). While the former is typically unrealistic on any real network, the latter again misses the underlying path structure.

Third, the naïve approach of recomputing an exact shortest path for every evaluation requested by the vehicle routing algorithm. This approach is scarcely used due to its computational expensiveness, despite it is the one that formally proves to return always the most reliable travel time estimate. Unfortunately, this proved not to be feasible due to the massive amount of typical travel time requests (millions of evaluations per second).

All the methods above must be further complicated, considering that the travel time information changes over time. The common approach of using travel times known in advance is at the end not practicable, also because a crucial factor must be considered in the design of a good-quality service delivered in the real world: the intrinsic within-day variability of traffic congestion is affected by unexpected network disruptions and mutable weather conditions. The best possible forecasts should then be constantly updated.

Nomenclature

l	size of the cells of the spatial grid
n	number of the reference points
Δv	time discretization of the network item performances
Δt	time interval between departure time instants of travel time matrices
W	time window on which travel time matrices are computed
r	number of travel time matrices to cover time window W
Δu	time interval of refresh of the travel time matrices

2. Methodology

In this paper we present a complete methodology to achieve an extremely fast computation of point-to-point travel times based on a dynamically updated forecast of traffic conditions on a large network. With respect to the abovementioned classifications, this could be summarized as a known matrix of point-to-point travel times amongst reference points randomly chosen, whose travel time estimates are periodically updated by an exact dynamic shortest path algorithm.

2.1. The infomobility data

A plurality of big data streams concerning traffic information is exploited. Link speeds from loops, cameras or FCD provide a real-time information about the network elements at the current time (in real applications "real-time" typically means continuously updated with a frequency between 1 and 15 minutes). This information is common in traffic management control centers, which act as data hubs, collecting the data from proprietary sensors over the network. The IoT is making this data more and more available, as open data or free creative common license (see http://opentransportmap.info/).

Additionally, other data streams such as vehicle counts can be used as an input to refresh the network forecast at given time intervals (e.g. every 5 minutes), combining model-based and statistical approaches, as described in Meschini *et al.* (2010). This provides a forecast of the network item travel times in the near future. A common span of such forecast is one hour in the future, but it can be extended to cover the time window of the paths of our use-case.

The combination of these two items provides a reliable source of data about travel times along network items both in the current and in the future instants. This covers the needs of the typical vehicle routing and journey planning applications.

2.2. The reference points

Now, the network area is ideally partitioned by a spatial grid of cells of size l. A given number n of reference points are (randomly) selected from inside the grid. There is no constraint for such points: they can be nodes of the graph, points of interest, coordinate points, etc.

Every grid cell is univocally associated to the closest reference point, for spatial indexing. This detail will be used again later.

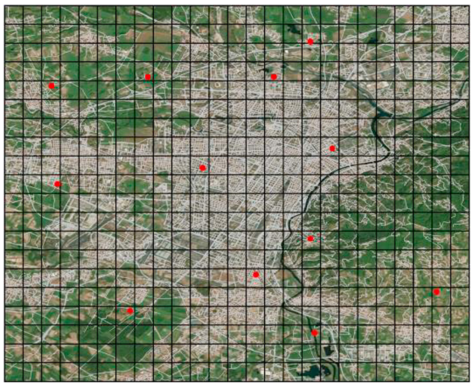

Figure 1: The study area, partitioned by a spatial grid and some of the reference points.

2.3. The shortest path estimate

The dynamic shortest path algorithm presented in Attanasi *et al.* (2013) computes the shortest path (and then minimum travel time) on the network starting at given instants amongst all the reference points. The used algorithm is dynamic: it considers the shortest path search trajectory in the space and time and then fully copes with the dynamicity of the traffic phenomena. The algorithm is a generalized implementation of Pallottino *et al.* (1998). Plus, it exploits all the data mentioned at paragraph 2.1: the solution it returns is then consistent with the dynamic data (events, measured traffic states, forecasted traffic states) available before the computation start.

To make the shortest path algorithm maximally performant, an additional optimization has been implemented, discretizing time intervals. First, the travel time of every network link is computed per every entry instant over the optimization window W, with period Δv. Then, the dynamic shortest path search uses as travel time of every link met during the exploration an interpolated travel time from the computed periodical performances, as shown in Figure 2 below.

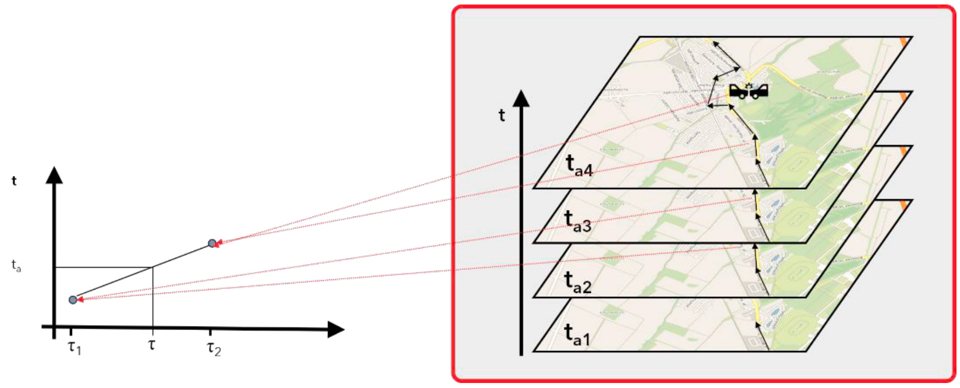

Figure 2: The travel time of a link given the entry time τ is computed as the interpolation between the pre-estimated link travel times at the previous and successive entry time instants τ_1 and τ_2.

As a result of the whole process, a matrix of travel times amongst origin-destination reference points is computed, as described in the procedure of the one-to-all minimum cost paths in Chabini (1998). It is relevant to highlight that despite the algorithm is extremely efficient and additionally designed to compute at once the tree of all the dynamic shortest paths departing from one point at one desired time interval (one row of the matrix), this computation is separable and can further scale via distributed computing.

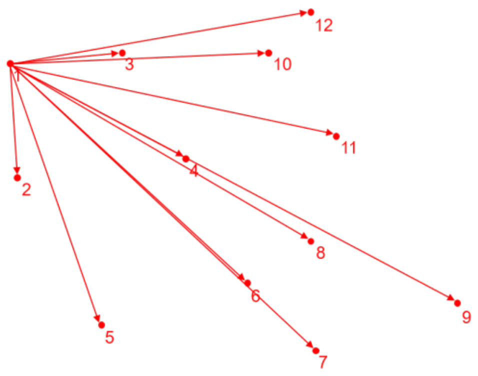

Figure 3: A shortest path tree amongst reference points.

Table 1. The travel times corresponding to the shortest paths in Fig. 2.

o\d	1	2	…	n
1	T_{11}	T_{12}	…	T_{1n}
2				
…				
n				

Furthermore, travel time matrices are computed per a set of r departure time instants, every Δt time, to cover the time window W on which to optimize the vehicle routing, as shown in Figure 4 below. The abovementioned parallelization can then be applied for departure time instants, too.

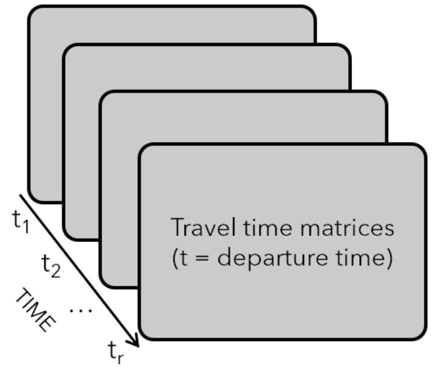

Figure 4: Travel time matrices are computed per periodic departure time instants.

2.4. Usage of the travel time matrix information

The resulting dynamic od matrices of travel times among reference points is then used, in combination with the spatial indexing, to provide a quick estimation of point-to-point travel times based on geometrical considerations. In particular, given any two points A and B, the travel time between them is estimated via the formula:

$$T_{AB} = T_{12} \cdot \frac{d_{A1}+d_{12}+d_{2B}}{d_{12}} \qquad (1)$$

where 1 and 2 are the reference points associated respectively to A and B via the spatial indexing.

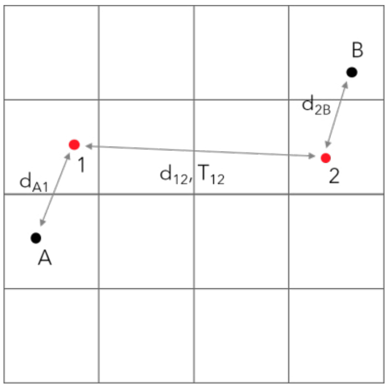

Figure 5: The travel time between any two points is estimated based on geometrical assumptions.

Equation 1 reproportions the travel time between the two reference points 1 and 2, which was explicitly computed and is then highly reliable, onto the two additional segments estimated to be travelled to go from A to B, of length dA1 and d2B. The rationale behind equation 1 is linked to the idea that connectivity on road networks is not diffuse. Then, given the path between two points (in our case 1 and 2), the path to connect two terminal points A and B respectively close to 1 and 2 is better approximated assuming 1 and 2 as waypoints, rather than just using the distance as the crow flies from A to B.

The proof of the assertion above is out of the scopes of this paper, as the formula to reproportion T12 to the distance between A and B gives a minor contribution to the effectiveness of the method.

More accurate estimations of the actual travel time estimated between two path terminal points, knowing the one between the two relevant points, can of course be provided. An example is described in Yang *et al.* (2018). That implies to have defined a detour ratio coefficient, which allows to compute the true travelled distance (and time), based on some additional information about the couple of the two terminals. Some typical information considered are the distance as the crow flies, area, projection link hierarchies. Nevertheless, this approach requires a preliminary study and calibration related to the application area, while equation 1 represents a rough but effective and efficient estimate which has the strong pro of identically holding for any network.

A further work could estimate the improvements of effectiveness of more complex formulas with respect to equation 1, but as stated above, this is not the key contribution of the described methodology.

2.5. Update of the travel time information

Finally, to ensure the consistency with the dynamically updated data of paragraph 2.1, the computation of the travel time matrices is periodically performed every Δu. This ensures that the system is reactive in at most Δu time to any real-time event and any related estimated subsequence via the traffic state forecast engine.

3. Results

The quality of the fast estimates has been validated on the real network shown in Figure 1, covering the center of Turin, a north Italy city. The network size is approximately 20 x 25 km, for about 100.000 link items and 850.000 km of roads covered.

The first validation which has been done is to compare the approximated dynamic shortest path (ASP in reported figures) described in paragraph 2.3 against the slower but accurate output of the original journey planner software (without the link performance precomputation, HP in reported figures), given that both are using the same knowledge base of network attributes and dynamic traffic conditions.

In the test results below, we had the configuration described below.

Table 2. The fixed configuration parameters used for the sensitivity analysis of other parameters.

Parameter	Value
Δv	5 min
Δt	15 min
W	3 hours
r	12
Δu	5 min

The chart below shows the regression between the two shortest paths, in the case reference points are randomly selected from nodes of the network (left) or coordinate points in the space (right). The former yields a relative error of 0.22%, while the latter 7.68%.

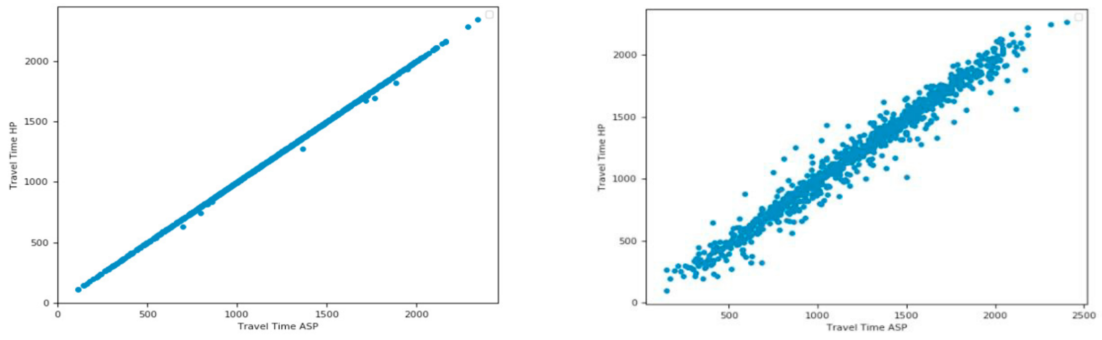

Figure 6: Regression chart between the exact dynamic shortest path (HP) and the one approximated via network item performance precomputation (ASP) in the cases reference points are randomly selected from nodes of the network (a) or coordinate points in the space (b).

The outliers of case (a) have been singularly checked and it has been verified that these only depend on a different search starting point criterion: the ASP uses the closest reference point while HP allows alternatives, where these have the same distance with respect to the input starting point. In fact, differences occur when the graph has overlapping nodes, because HP can choose the best possibility, while the ASP just chooses the one which was the reference point.

This difference increases when dealing with reference points random in space, because the most accurate logics of placing of the initial point make a greater difference with respect to the much rougher approximation of the grid spatial indexing. All other point exactly coincide, with the only other error due to the interpolation of the precomputed link performances.

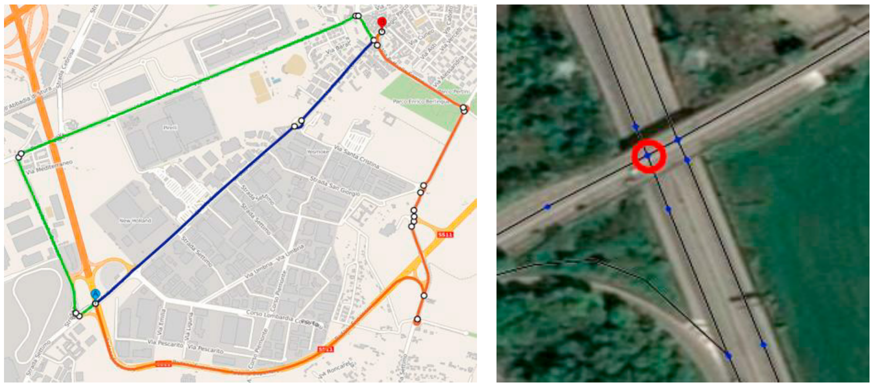

Figure 7: An example of different paths between HP (blue) and ASP (brown) (a) and a detail of the graph, in which two nodes overlap for two roads on different levels: the highway and the underlying minor road (b).

Then we performed a sensitivity analysis of the variation of the error by changing some other configuration parameter. First, how the error changes with respect to the number of relative points n. It can be noticed that of course the error decreases by increasing n, but this has a smaller impact once reached a given value.

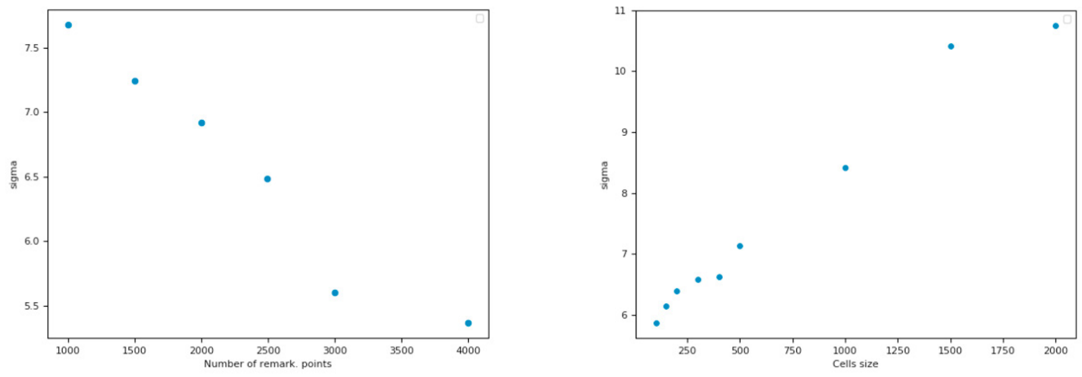

Figure 8: Sensitivity analysis of the relative error with respect to the number of reference points (a) and the grid cell size (b).

This is reasonable as once reached a quality threshold due to the reference point density, increasing the number of points does not add value to the result of the indexing. On the other side, increasing the cell size l obviously makes the spatial indexing rougher and rougher, so the smallest the grid the more accurate the spatial index association.

Finally, we highlight that the response times of such an approach are in the order of magnitude of a look-up table (1 millionth second), with a minimum response time of $1.85 \cdot 10^{-6}$ s and a maximum of $3.25 \cdot 10^{-6}$ s.

The machine running the algorithm was a 2.4 GHz x 4 cores CPU, 16 GB RAM.

4. Conclusions

The numerical analysis showed a very favourable trade-off between accuracy loss and efficiency gain, thus making the proposed approach feasible for any service operations. To our knowledge no other approach aimed to fulfill in a real working environment both so efficient computational times and reliable results, further addressing the responsiveness to real world network condition changes.

The hardware required to run the components is far inside the commonly available PC characteristics, without requiring outstanding server hardware. The number of machines needed to run in parallel the travel time matrices update is definitely in the possibilities of any provider of such advanced service.

Further refinements on this work could be reasonably spent in improving the starting point logics, to further decrease the error in the case of points in space as reference points. Also, an analysis of the responsiveness and effectiveness in presence of real-time dynamic data could be specifically carried out.

Acknowledgements

We thank our colleagues Andrea Procaccini, Alessandro Rotundo and Edmondo Silvestri from PTV Group - SISTeMA who provided insight and expertise that greatly assisted the research.

References

Agarwal, P., Varadarajan, K., 2004. A near-linear constant-factor approximation for euclidean bipartite matching?. Proceedings of the Twentieth Annual Symposium on Computational Geometry - SCG 04.

Attanasi, A., De Cristofaro, S., Meschini, L., Gentile, G., 2013. Hyperpath journey planner: a dynamic shortest pathfinder for multimodal transportation networks. EURO INFORM 2013 Proceedings.

Brandão, J., 2011. A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem. Computers & Operations Research, 38 (1), 140-151.

Chabini, I., 1998. Discrete Dynamic Shortest Path Problems in Transportation Applications: Complexity And Algorithms With Optimal Run Time. Transportation Research Records, 1645(1), 170–175.

Jung, J., 2012. A Simulation Model for Evaluating Demand Responsive Transit: Real-Time Shared-Taxi Application. Journal of the Korean Society of Road Engineers, 14(3), 163-171.

Meschini L., Gentile G., 2010. Real-time traffic monitoring and forecast through OPTIMA – Optimal Path Travel Information for Mobility Actions. MT-ITS 2009 Proceedings of Models and Technologies for Intelligent Transportation Systems.

Miller, C. E. , Tucker, A. W., Zemlin, R. A., 1960. Integer Programming Formulation of Traveling Salesman Problems. Journal of the ACM 7.4, 326-329.

Pallottino, S., Scutellà, M.G., 1998. Shortest Path Algorithms in Transportation Models: Classical and Innovative Aspects. Equilibrium and Advanced Transportation Modelling, ch. 11.

Yang, H., Ke J., Ye J., 2018. A universal distribution law of network detour ratios. Transportation Research Part C: Emerging Technologies, 96, 22-37.