Fast Traffic Assignment by Focusing on Changing Edge Flows (Extended Abstract)

Ali Davoodi,[1] **Mark Wallace,** [1] **Daniel Harabor** [1]

[1] Faculty of Information Technology, Monash University, Melbourne, Australia
ali.davoodi,mark.wallace,daniel.harabor@monash.edu

Abstract

This paper presents a novel algorithm for solving the traffic assignment problem (TAP). Contrary to traditional algorithms, which use the one-to-all shortest path algorithm to solve the problem for all origin destinations (OD) pairs, this algorithm tracks the changes of the edges and (at certain iterations) solves the problem only for critical edges whose flows have changed substantially using a state-of-the-art edge p2p shortest path algorithm. When additionally, only OD pairs with larger flows are considered, this enhancement halves the time needed to optimize the solution with a very small error in a large-scale network.

Introduction

Traffic assignment is how travellers choose their routes in a network where the demands between origins and destinations are given. Wardrop's first principle (Wardrop and Whitehead 1952) states that for each pair of origin-destination (OD), the travel time of the used routes (positive flow) is equal to or less than the travel time of the unused routes. Traffic flows that establish this rule are referred to as "equilibrium flows". (Beckmann, McGuire, and Winsten 1955) formulated the TAP as an optimisation problem based on the UE condition. Since then, many algorithms have been introduced to solve the traffic assignment problem. See (Mitradjieva and Lindberg 2013) for a recent overview. These algorithms have an iterative method. Current solutions and shortest paths are used in each iteration to shift flow from non-shortest paths to shortest paths.

Calculating the shortest paths for each pair of OD in each iteration is one of the most important and time-consuming components of traffic assignment algorithms. Instead of using the p2p shortest path technique, traffic assignment algorithms often employ the shortest path tree such as Dijkstra (Dijkstra et al. 1959) , or Bellman-ford (Bellman 1958) one-to-all from each origin to all nodes to find the shortest paths. The reason for this is that in these classical algorithms, searching one-to-all shortest paths is not substantially more costly than p2p shortest paths. Despite numerous acceleration approaches and tactics that have been researched to reduce shortest path time computation (readers referred to (Bast et al. 2016) for a recent overview), classical one-to-all shortest path algorithms are still difficult to beat.

In this paper, instead of solving the problem for all OD pairs, the problem is solved only for the OD pairs that have more potential to reduce the objective function. In this study, we solve TAP using (Chen, Jayakrishnan, and Tsai 2002)'s Origin-Destination-Based Frank-Wolfe (ODBFW) algorithm. This path-based algorithm breaks the problem down into sub-problems for each OD pair and solves each sub-problem using the FW algorithm. Compressed Path Databases (CPDs)(Bono et al. 2019), a p2p technique, is also used to find the shortest paths. The critical OD pairs can be identified by tracking edge changes and storing the OD pairs associated with each edge. The convergence of the proposed algorithm is directly compared to the conventional ODBFW, as well as RG (Babazadeh et al. 2020) algorithm.

Problem Setting

Consider $G = (N, E)$ is a directed transportation network with edge set E and node set N. Each edge $e \in E$ has a non-decreasing and continuously differentiable travel time function $t_e(x_e)$ where x_e shows flow of edge e. R and S are denoted as sets of origins, and destinations respectively. D_{rs} is demand from origin r to destination s. The sets of paths from origin r to destination s is defined P_{rs}, and h_p considering as the flow on path p. The traffic assignment problem based on Wardrop's first principle can be formulated as a non-linear programming problem (NLP) with strictly convex and twice continuously differentiable objective function and linear constraints as follows:

$$min\, z(x(h)) = \sum_{e \in E} \int_0^{x_e} t_e(\omega)\, d\omega \tag{1}$$

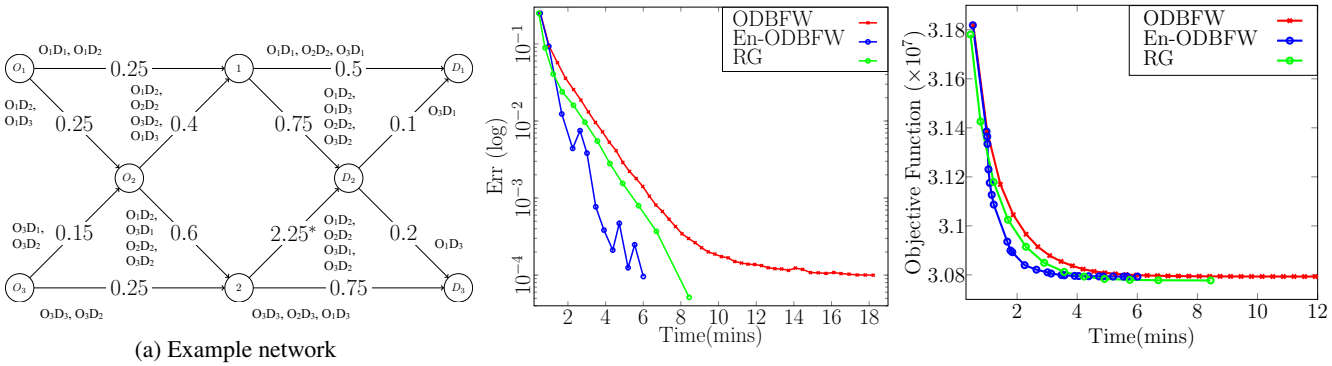
$$\sum_{p \in P_{rs}} h_p = D_{rs} \qquad \forall r \in R, \forall s \in S \tag{2}$$

$$h_p \geq 0 \qquad \forall p \in P_{rs}, \forall r \in R, \forall s \in S \tag{3}$$

and the relation between edge flows and path flows is expressed as:

$$x_e = \sum_{r \in R} \sum_{s \in S} \sum_{p \in P_{rs}} h_p \delta_{ep} \qquad \forall e \in E \tag{4}$$

Where $\delta_{ep} = 1$ if path p passes edge e, and 0 otherwise.



(a) Example network

(b) Convergence rates

(c) Objective function values

Figure 1: Performance comparison of ODBFW and En-ODBFW for Chicago network

Proposed Algorithm

Our proposed algorithm is based on the fact that in each iteration of the TAP algorithms, the OD pairs that undergo more changes are those whose paths include edges that had more changes in the previous iteration. Thus, the aim of our algorithm is to track the edges that have experienced the most changes in their edge cost function $EC_e = x_e.t(x_e)$ to solve the sub-problem only for the OD pairs that have used these edges through their path. To identify related OD pairs of each edge, we first implement the regular ODBFW cycle to reach acceptable solution accuracy. Then, the algorithm stores the OD pair number of each used edge by moving through the paths. Edges whose cost difference in the current iteration compared to the previous iteration are greater than the threshold changes will be considered effective edges. The number of effective OD pairs is subsequently discovered. If this number is too large (it exceeds a given threshold), then the Dijkstra one-to-all shortest path algorithm is used to solve all OD pairs' sub-problems. Otherwise, the problem for effective OD pairs is solved using the p2p CPD-search shortest path. Finally, we increase the algorithm's sensitivity for tracking edges by halving the threshold changes. For example, consider Figure 1a with 9 OD pairs. In this figure, for each edge, ΔEC_e is written on it. The OD pairs passing through each edge are also specified next to each edge. Since edge (2,D_2) has experienced the most changes in flow and travel time in the previous iteration, the OD pairs passing through it have higher potential to reduce the objective function than the other OD pairs. So, in this iteration, our algorithm targets 4 OD pairs passing edge (2,D_2) instead of solving the problem for all 9 OD pairs.

Storing all OD pairs for edges takes a large amount of memory. Also, solving sub-problems for many OD pairs may not be very effective due to the low demand. To address this issue, we can only store those with sufficient demand $D_{rs} \geq D_{th}$ and let the rest be solved in regular iterations. Our research shows that using this strategy cuts the time to solve the problem in half.

Numerical Results

This section compares the performance of the proposed En-ODBFW algorithm to ODBFW and RG algorithms to investigate its performance. The authors wrote these algorithms in the C++ programming language using Visual Studio code. Both algorithms were implemented on a Mac Pro 13 with 16 GB of RAM. The Chicago regional network was chosen as a large-scale network with 1790 zones and more than 3 million OD pairs. We run En-ODBFW algorithm with threshold demand $D_{th} = 1$. The algorithm's convergence is assessed by the Err, which is the relative difference between the current total cost and the optimal total cost in the current flow. Our runs were terminated when Err fell below 10^{-4}.

Figure 1b compares the performance of ODBFW, RG, and En-ODBFW for the Chicago networks by plotting their Errs versus CPU times. While 1c shows how quickly these algorithms can reduce the Beckman model's objective function value. These figures show that En-ODBFW clearly outperforms the ODBFW algorithm. ODBFW took around 18.2 minutes to achieve $Err = 10^{-4}$, while the En-ODBFW algorithm converged to the same precision in less than 8 minutes, which is nearly 2.3 times faster than ODBFW. These figures also reveal that En-ODBFW can be faster than the RG algorithm to attain the target convergence and hits the plateau faster than this state-of-the-art algorithm.

Conclusion

This study proposes a novel variation of the Frank-Wolfe (FW) traffic assignment algorithm. While the shortest path tree has provided the basis for the fast recent FW variants, this paper has shown that processing single origin-destinations (OD) can be even faster. Our algorithm intelligently targets ODs which most impact the objective, when there are not too many of them. Otherwise, it falls back on standard path tree based iterations. On typical examples computing time is better than halved. Our variation can use the best available shortest path methods, and benefit from any future enhancements.

References

Babazadeh, A.; Javani, B.; Gentile, G.; and Florian, M. 2020. Reduced gradient algorithm for user equilibrium traffic assignment problem. *Transportmetrica A: Transport Science*, 16(3): 1111–1135.

Bast, H.; Delling, D.; Goldberg, A.; Müller-Hannemann, M.; Pajor, T.; Sanders, P.; Wagner, D.; and Werneck, R. F. 2016. Route planning in transportation networks. In *Algorithm engineering*, 19–80. Springer.

Beckmann, M. J.; McGuire, C. B.; and Winsten, C. B. 1955. *Studies in the Economics of Transportation*. Rand Corporation.

Bellman, R. 1958. On a routing problem. *Quarterly of applied mathematics*, 16(1): 87–90.

Bono, M.; Gerevini, A. E.; Harabor, D. D.; and Stuckey, P. J. 2019. Path Planning with CPD Heuristics. In *IJCAI*, 1199–1205.

Chen, A.; Jayakrishnan, R.; and Tsai, W. K. 2002. Faster Frank-Wolfe traffic assignment with new flow update scheme. *Journal of Transportation Engineering*, 128(1): 31–39.

Dijkstra, E. W.; et al. 1959. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1): 269–271.

Mitradjieva, M.; and Lindberg, P. O. 2013. The stiff is moving—conjugate direction Frank-Wolfe Methods with applications to traffic assignment. *Transportation Science*, 47(2): 280–293.

Wardrop, J. G.; and Whitehead, J. I. 1952. Correspondence. some theoretical aspects of road traffic research. *Proceedings of the institution of civil engineers*, 1(5): 767–768.