Computationally-Efficient Decomposition Heuristic for the Static Traffic Assignment Problem

Venktesh Pandey[1], Priyadarshan N Patil[2], Manisha Ganesh[3], and Stephen D Boyles[1*]

1 Introduction

Traffic assignment problems (TAPs), which assign vehicles to network routes as the last step of the four-step planning process, are well studied. With advances in computing power, several efficient algorithms have been proposed such as bush-based algorithms (Nie 2010). This research is motivated by the further need to improve the computational efficiency of solving TAPs on large-scale networks.

The first motivation is the planning for megaregions. Megaregions cross state and political boundaries and are characterized by trade and infrastructural connections across a much larger area. For example, the north-eastern megaregion spans the states of Connecticut, Rhode Island, New York, Massachusetts, Pennsylvania, Delaware, New Jersey and Virginia, and contains as much as 17% of the current US population. With increasing intra-megaregion trade and traffic, current statewide or citywide models are not appropriate for the megaregion scale creating a need for computationally-efficient TAP algorithms for these networks.

The second motivation is rooted in applications that require multiple solution of TAPs such as scenario analysis and network design problems. These applications can benefit significantly from faster TAP algorithms. Furthermore, for such applications we often desire only an approximate solution to the TAP, characterized by a higher *relative gap* value, than an optimal solution at which the relative gap is zero (Patriksson 2015).

Lately, network decomposition algorithms have been proposed for the TAP (Jafari et al. 2017, Raadsen 2018). These algorithms decompose a large network into subnetworks which can be solved in parallel and offer a potential to outperform algorithms that solve TAP on the full network (referred as centralized algorithms) by making use of multiple machine cores. However, scaling these algorithms for more than two network partitions and handling subnetwork interactions is challenging. In this article, we propose a decomposition-based heuristic that generates approximate solutions to the TAP in faster computation time than the centralized algorithms.

The contributions of this research include the following: (a) we propose a network transformation to solve a restricted shortest path (SP) problem as a subproblem of the heuristic, and (b) we propose an iterative refinement approach for network partitioning that improves the computational efficiency of the heuristic. The rest of the abstract is organized as follows. Section 2 presents the heuristic and discusses the component problems, while Section 3 reports experimental results on different test networks and provides a brief discussion of the findings.

2 Methodology

2.1 DSTAP vs DSTAP-Heuristic

Jafari et al. (2017) proposed a decomposition algorithm for static traffic assignment problem (DSTAP) where a full network is replaced by geographically separate regions called subnetworks and a master network that approximates the interactions between subnetworks by connecting origin, destination, and boundary nodes with artificial links whose parameters are estimated using TAP sensitivity analysis. Figure 1(b) shows the DSTAP master network and subnetworks obtained after decomposing the full network in Figure 1(a). It was shown that iteratively solving the master network and subnetworks, while updating the parameters of artificial links, is guaranteed to converge to the optimal TAP solution with additional computational savings for large-scale networks.

Implementing DSTAP for megaregions and other large-scale networks is advantageous as it distributes the computations over subnetworks. However, the number of artificial links generated pose a key computational bottleneck.

*Emails: venktesh@utexas.edu (corresponding author), priyadarshan@utexas.edu, manisha.ganesh@utexas.edu, sboyles@mail.utexas.edu
[1]Department of Civil, Architectural and Environmental Engineering, The University of Texas at Austin
[2]Department of Operations Research and Industrial Engineering, The University of Texas at Austin
[3]Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin

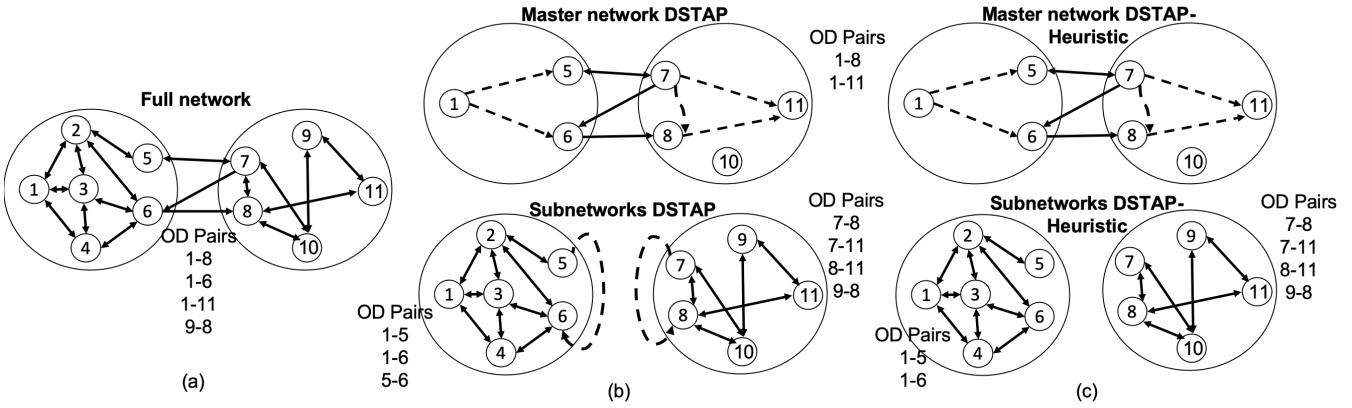


Figure 1: (a) Full network, (b) DSTAP networks, and (c) DSTAP-Heuristic networks

Subnetwork artificial links (SALs) such as links (5,6) and (7,8) in subnetworks in Figure 1(b), which represent all the paths connecting the two nodes completely outside the given subnetwork, are particularly challenging due to two reasons. First, at any DSTAP iteration, a path replacing an SAL may itself include SALs in other subnetwork, which adds an unnecessary complexity to the mapping of flow on the full network. Second, as discussed in (Jafari et al. 2017, Appendix B), modeling SALs for more than two subnetworks requires approximating a part of the master network in all other subnetworks which can be time consuming.

In this research, we modify DSTAP by simply dropping the SALs (networks in Figure 1(c)). Because we exclude SALs, the convergence of DSTAP is no longer guaranteed and DSTAP now becomes a heuristic which we call DSTAP-Heuristic. Algorithm 1 shows the steps for DSTAP-Heuristic in detail.

Algorithm 1 DSTAP-Heuristic for solving TAP

Inputs: network file, trips file, networks partitions as subnetworks
Step 1: Set up — validate partitions for connectivity, create master networks and subnetworks, generate master net artificial links and corresponding artificial origin-destination (OD) pairs inside subnetworks.
Step 2: Optimize
while relative gap on the full network > threshold AND iteration number < desired maximum iterations **do**
    Solve master network to a desired gap
    Update the demand on subnetwork artificial OD pairs
    Solve the subnetworks in parallel with the updated demand to a desired gap
    Update the parameters for master network artificial links
Outputs: Final full network gap and link flows/costs

2.2 Constrained shortest path problem for the master network

A path in the master network must not include two consecutive artificial links. This is because, if there were two consecutive artificial links x and y, then the subpath $[x, y]$ effectively models all paths within a subnetwork connecting the head node of x (x_h) with the tail node of y (y_t); however, these paths should instead be modeled through a direct artificial link connecting x_h and y_t.

Standard label-correcting algorithms cannot be used to find SP under this constraint as the Bellman property is not satisfied. As an example, consider the master network shown in Figure 2(a) where both nodes 8 and 11 are destinations in subnetwork 2. The SP to node 11 which does not contain two consecutive artificial links is $[1, 5, 7, 6, 8, 11]$; however, the subpath $[1, 5, 7, 6, 8]$ is not the SP to node 8.

To solve the constrained SP problem, we propose a network transformation where all destinations in the subnetwork not containing the origin are split into dummy nodes, one for each of its incoming link. Then, instead of the original destination, these dummy nodes are connected to the corresponding incoming link. The outgoing links from dummy nodes are created for all outgoing links from the original destination except if both the outgoing and the incoming links are artificial. Lastly, each dummy node is connected to the original destination using zero-cost links. Figure 2(b) shows how node 8 is split into dummy nodes 87 and 85 modeling the incoming links $(7, 8)$ and

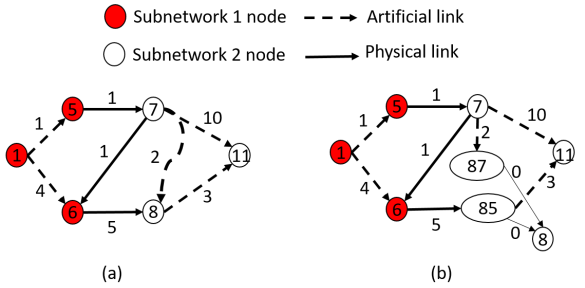


Figure 2: (a) Original master network, and the (b) transformed master network

$(5, 8)$, respectively. We prove that using standard label-correcting algorithms on the transformed network solves the constrained SP problem. The details are skipped for brevity.

2.3 Partitioning algorithms

Computational performance of DSTAP and DSTAP-Heuristic depends on the choice of subnetwork boundaries. A good partition is the one that creates fewer boundary nodes and where the subnetworks are "balanced" in size and have minimal "interactions" among each other. Yahia et al. (2018) compared two partitioning algorithms: the SDDA algorithm, which incrementally builds the subnetworks using seed nodes located far apart, and a flow-weighted Spectral partitioning algorithm that minimizes the flow across all cut arcs (assuming the equilibrium flows are available). While both algorithms can be applied for the DSTAP-Heuristic, we desire partitions that minimize the flows leaving the subnetwork and entering back again as this minimizes the role of SALs providing better convergence.

We define a statistic ψ as the difference between the total flow across the cut arcs (interflow) and the total demand for OD pairs with origins and destinations in different subnetworks (interdemand). If at equilibrium no path between an OD pair leaves a subnetwork and enters it back again (implying zero flow on all SALs), $\psi = 0$. We hypothesize that partitions with lower value of ψ generate solutions with lower relative gap than the ones with higher ψ.

We propose an iterative partitioning-refinement algorithm which is based on refinement approach proposed in Fiduccia and Mattheyses (1982) (commonly referred as the FM refinement). Given an initial partition, the algorithm performs local improvement by evaluating the boundary nodes and moving one of the nodes that results in the highest reduction in the current value of ψ, until a fixed number of iterations or until no refinement is possible. The details of the ψ-FM algorithm are similar to the FM algorithm and are skipped for brevity.

3 Selected Results And Discussion

We conduct experiments on various test networks obtained from Transportation Networks for Research Core Team (2019). For each experiment, we consider the ψ-FM refinement with different initial partitions including SDDA, Spectral, and the partitions generated by the well-known partitioning solver METIS (Karypis and Kumar 1998).

Table 1 shows the results for different test networks. Column 2 shows the lowest relative gap obtained from running DSTAP-Heuristic and the partition obtaining that gap. Columns 3 and 4 show the computation time taken by DSTAP-Heuristic and the centralized algorithm to reach that gap (using the gradient projection algorithm as the TAP solver for all cases). The computation times are reported in seconds as observed on a 3.3GHz Windows machine with 8 GB RAM. The rows in the table are ordered in increasing size (number of nodes) of the test network.

We make following key observations:

1. Convergence level of DSTAP-Heuristic depends on the choice of partitioning algorithm. The best achieved gap is in the range of 0.23–9.44E-05 for various networks.

2. There is no consistently superior partitioning algorithm that works well for all networks indicating that there is no one-size-fits-all solution. However, we observe that the ψ-FM refinement does better than the partitions without this refinement for 6 out of the 8 tested networks.

3. DSTAP-Heuristic is slower than the centralized algorithm for smaller network instances. However, for larger networks including Austin, Goldcoast, Philadelphia, and Chicago Regional, DSTAP-Heuristic takes lesser time generating a percent time savings ranging between 2.51–62.6%. Figure 3 shows the partitions on these networks.

Table 1: Comparison of computation time for generating solutions up to a certain relative gap using DSTAP-Heuristic vs centralized algorithm (GP=gradient projection)

		Computation time till best gap (sec)		
Network	**Best gap achieved using DSTAP-Heuristic**	**DSTAP-Heuristic using GP**	**Centralized using GP**	**%-savings**
SiouxFalls	9.44E-05 (ψ-FM-METIS)	2.37	**0.16**	NA
Eastern Massachusetts	3.91E-04 (ψ-FM-Spectral)	0.59	**0.16**	NA
Barcelona	0.0036 (Spectral)	4.26	**2.06**	NA
Chicago Sketch	0.0039 (Spectral)	13.27	**8.12**	NA
Austin	0.233 (ψ-FM-METIS)	**135.84**	185.84	26.9%
Goldcoast	0.063 (ψ-FM-Spectral)	**113.07**	302.12	62.6%
Philadelphia	0.027 (ψ-FM-METIS)	**702.36**	720.46	2.51%
Chicago Regional	0.062 (ψ-FM-METIS)	**854.39**	1305.95	34.57%

While the best achieved relative gap is still high, the computational advantage of DSTAP-Heuristic makes it suitable for obtaining an approximate TAP solution or for warmstarting a TAP using standard algorithms.

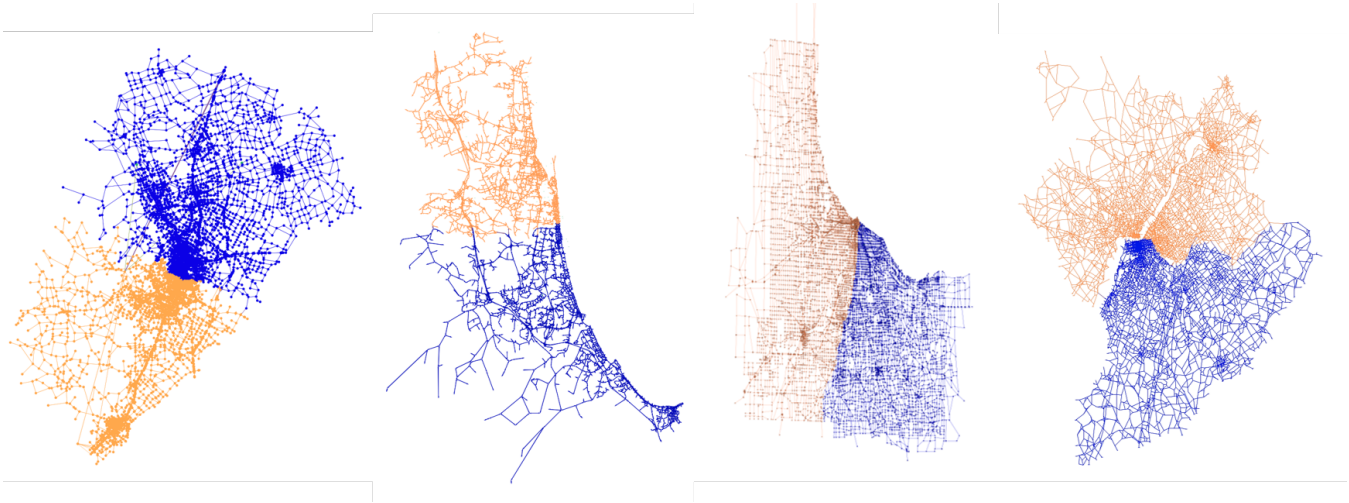


Figure 3: Partitions for the (a) Austin, (b) Goldcoast, (c) Chicago Regional, and (d) Philadelphia networks

References

C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *19th Design Automation Conference*, pages 175–181. IEEE, 1982.

E. Jafari, V. Pandey, and S. D. Boyles. A decomposition approach to the static traffic assignment problem. *Transportation Research Part B: Methodological*, 105:270–296, 2017.

G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1):359–392, 1998.

Y. M. Nie. A class of bush-based algorithms for the traffic assignment problem. *Transportation Research Part B: Methodological*, 44(1):73–89, 2010.

M. Patriksson. *The traffic assignment problem: models and methods*. Courier Dover Publications, 2015.

M. P. H. Raadsen. *Aggregation and decomposition methods in traffic assignment: towards consistent and efficient planning models in a multi-scale environment*. PhD thesis, University of Sydney, 2018.

Transportation Networks for Research Core Team. Transportation Networks for Research. `https://github.com/bstabler/TransportationNetworks`, 2019. Accessed: December 15, 2019.

C. N. Yahia, V. Pandey, and S. D. Boyles. Network partitioning algorithms for solving the traffic assignment problem using a decomposition approach. *Transportation Research Record*, 2672(48):116–126, 2018.