A Comparative Analysis of Heuristics for the Improved Convergence of Dynamic Traffic Assignment Models

Undergraduate Honors Thesis

Michael Levin
Network Modeling Center
The University of Texas at Austin
michaellevin@utexas.edu

Supervisor: Dr. Stephen Boyles

Presented to the Department of Computer Science of the University of Texas at Austin in partial fulfillment of the requirements for the degree of BACHELOR OF SCIENCE IN COMPUTER SCIENCE (TURING OPTION)

December 2012
A Comparative Analysis of Heuristics for the Improved Convergence of Dynamic Traffic Assignment Models

Undergraduate Honors Thesis

Michael Levin
Network Modeling Center
The University of Texas at Austin
michaellevin@utexas.edu

Supervisor: Dr. Stephen Boyles

Abstract

Simulation-Based Dynamic Traffic Assignment (DTA) provides a high level of detail useful in realistic modeling for transportation planning. However, mathematical optimization of the solution algorithm is difficult because of discretized space and time, and as a result solution algorithm computation times measured in hours or even days can discourage use. This work compares and analyzes experimental results from two types of potential techniques for reducing computation time: a static traffic assignment (STA) warm start to DTA, and existing and novel heuristics from variations on the
method of successive averages to gradient-based methods similar to those used in STA. Warm start results indicate the potential for increases in time and possibly solution convergence. Some DTA solution heuristics improve convergence significantly in a small network, suggesting the potential for benefit to larger networks. Further research may result in either technique becoming viable for practitioners.
Acknowledgements

I am extremely grateful for the efforts of my advisor, Dr. Stephen Boyles, for his guidance through the numerous revisions and drafts. I would also like to thank Dr. Nezamuddin and Dr. Natalia Ruiz-Juri, with whom I worked on the static warm-start techniques; and Matt Pool, Travis Owens, and Dr. Natalia Ruiz-Juri, with whom I worked on the Method of Successive Averages heuristics. Finally, I would like to thank Dr. Jennifer Duthie for her advice on this and other projects at the Network Modeling Center; and Dr. S. Travis Waller, who introduced me to network modeling research.
## Contents

1 Introduction 1

1.1 Background 2

1.2 Static ‘Warm Starting’ 8

1.3 Heuristic improvements in the Method of Successive Averages 9

2 Methods of Static Warm Starting 11

2.1 Determining the initial path set 12

2.1.1 All-or-nothing paths 13

2.1.2 Paths within $\epsilon$ of the shortest path 14

2.1.3 Origin-Based Algorithm 15

2.2 Assignment of trips 15

2.2.1 Equal proportions 16

2.2.2 Density-based proportions 16

2.2.3 Origin-Based Algorithm 17

3 Analysis of Static Warm Starting 17

3.1 Test networks 17

3.2 Initial computational time 18

3.3 Convergence Results 19

3.4 Path usage 22

3.5 Artificial increase in congestion in the static User Equilibrium 23

3.6 Summary of warm-start results 25
Tables

Table 3.1 Test networks 18
Table 3.2 Computation time for finding paths 19
Table 3.3 Warm-start on Sioux Falls and Anaheim 20
Table 3.4 Path usage in Sioux Falls 22
Table 3.5 Warm-start on Sioux Falls: varying congestion in static 24
Table 4.1 Example time-varying cascade of step size values 29
Table 5.1 True gap comparison of all MSA-based heuristics 33
Table 5.2 True gap comparison of partial demand loading 34
Figures

Figure 1.1 Equilibrium Algorithms 5

Figure 1.2 Trip redistribution for the Method of Successive Averages 6

Figure 1.3 Visual Interactive System for Transport Algorithms (VISTA) convergence algorithm 8

Figure 1.4 Method of Successive Averages-based heuristics algorithm flow 10

Figure 3.1 Demand profile 18

Figure 3.2 % initial paths used in final solution plotted against cost gap % 23

Figure 5.1 Moving vehicles only from the least x% converged ODTs 35

Figure 5.2 Heuristics moving the worst vehicles 36

Figure 5.3 Demand moved in Sioux Falls and Anaheim by ODT vehsort 37

Figure 5.4 Gap distribution – % of gap from vehicles with an individual cost gap over 50% of individual travel time 38

Figure 5.5a Variations in step size (part a) 40

Figure 5.5b Variations in lambda (part b) 42

Figure 5.6 Simplicial Decomposition 43

Figure 5.7 New paths created by MSA and SD 44
1 Introduction

Traffic behavior models allow transportation planners to determine the effects of construction, accidents, or other changes to the roads or vehicle origins and destinations. Efficient methods of solving such models on a city-wide scale are necessary for analysis of the impact of proposed or ongoing changes. For instance, the network design problem of determining the placement and structure of new roads benefits from the ability to quickly evaluate traffic after new construction. Braess (1969) demonstrated that new roads can worsen travel time with Static Traffic Assignment (STA) link cost functions (see also Frank, 1981) and Daganzo (1998) similarly found that additional capacity added to bottlenecks in the more realistic spatial-queue flow model could also increase travel time for all vehicles.

Dynamic Traffic Assignment (DTA) models propagate flow more realistically than STA models at the cost of substantial increased computational time. Simulation-based DTA (SBDTA) models in particular track vehicle flow through traffic networks, making mathematical optimization of an objective function evaluated through simulation difficult. This paper develops and compares methods to reduce the computation time required to find an assignment close to the user equilibrium. Techniques are experimentally compared with currently used algorithms on two networks of different sizes to analyze effectiveness.
1.1 Background

A traffic network \( G = (N, A, V) \) is an extension of a directed graph, where \( N \) is the set of nodes (intersections and origins/destinations), \( A \) is the set of links (roads connecting nodes), and \( V \) is the demand or trips for the network. The set of zones, denoted by \( Z \), are a special class of node – origins and destinations – from which trips can originate. Formally, \( V \) is a matrix with entries \( v_{od} \) with \( o, d \in Z \) such that \( v_{od} \) is the number of vehicles departing origin \( o \) for destination \( d \). \( G \) is assumed to be connected for zones: for all \( o, d \in Z \), the set of all paths from \( o \) to \( d \), \( \Pi_{od} \), is non-empty. Each link \([i,j]\) has properties of length \( l_{ij} \), free flow (unobstructed) travel time \( \dot{c}_{ij} \), capacity (the maximum number of vehicles that can pass a point in unit time) \( q_{ij}^{\max} \), and the physical space limitation of jam density or maximum vehicles per unit length \( k_{ij}^{\max} \). Travel times \( c_{ij} \) are affected by link flow, denoted \( q_{ij} \), and for some flow models also the density \( k_{ij} \). Flow and density are determined by vehicle routing.

The traffic assignment problem for user equilibrium (UE) is to determine vehicle routing under the assumption that each driver selfishly selects the route to minimize personal travel time, resulting in an equilibrium in which no driver can improve his or her travel time by changing routes (Wardrop, 1952). Specifically, let \( c_{\pi} \) be the travel time for and \( f_{\pi} \) be the trips assigned to path \( \pi \). Then a network is in equilibrium if

\[
\forall o, d \in Z \forall \pi \in \Pi_{od} \left[ c_{\pi} = \min_{\pi' \in \Pi_{od}} (c_{\pi'}) \right] \forall \left[ f_{\pi} = 0 \right] \tag{1.1}
\]

At the UE, all used paths have travel time equal to the minimum per OD. The paths and assigned flow when UE is satisfied are a UE solution for the network, and, if the network
is properly calibrated, for the actual roads. In practice, models are solved to an acceptable excess cost or gap \( \delta \), a measurement of the distance from UE.

\[
\delta_{od} = \sum_{\pi \in \Pi_{od}} \left( f_{\pi} \left( c_{\pi} - \min_{\pi' \in \Pi_{od}} (c_{\pi'}) \right) \right) 
\]

\[
\delta = \sum_{o,d \in Z} \delta_{od} 
\]

From (1.1) and (1.2), at UE \( \delta = 0 \). Excess cost is often evaluated as a percentage of total travel time, denoted \( \hat{\delta} \), or as an average with respect to the number of trips, denoted \( \bar{\delta} \).

\[
\hat{\delta} = \frac{\delta}{\sum_{o,d \in \Pi_{od}} (f_{\pi} \times c_{\pi})} \times 100 
\]

\[
\bar{\delta} = \frac{\delta}{\sum_{o,d \in \Pi_{od}} v_{od}} 
\]

Depending on the flow model used to determine travel times, using mathematical programming methods to minimize \( \delta \) may be difficult.

STA models define \( c_{ij} \) to be a function of \( q_{ij} \). They have been extensively studied in the literature, and algorithms exist which can solve STA to minute gaps (such as \( \bar{\delta} < 1 \times 10^{-10} \)) in minutes for large city networks consisting of thousands of nodes and links (see Jayakrishnan, 1994; Bar-Gera, 2002; Dial, 2006). A major drawback, however, is their limiting flow model which ignores the effect of time-varying flows on links as well as congestion propagation across links due to limitations imposed by \( q_{ij}^{\text{max}} \) and \( k_{ij}^{\text{max}} \).

DTA models, based on the Lighthill-Whitham-Richards (LWR) model, developed by Lighthill and Whitham (1955) and independently by Richards (1956), address both of these issues. This work used a discretized version of the LWR model known as the Cell
Transmission Model (CTM), which was introduced by Daganzo (1994, 1995). CTM discretizes time into intervals of width $\Delta \tau$, and discretizes space by dividing links into a series of cells based on link length and free flow speed, such that a vehicle can traverse at most one cell during each simulation time interval. Specifically, link $[i, j]$ is divided into $\left\lceil \frac{c_{ij}}{\Delta \tau} \right\rceil$ cells. At free flow, flow requires $\left\lceil \frac{c_{ij}}{\Delta \tau} \right\rceil \times \Delta \tau$ to traverse the link. Flow moves through the cell graph, subject to cell limitations based on link properties. Flow is limited by capacity per time interval, $q_{ij}^{\text{max}} \times \Delta \tau$ as well as physical space through the cell jam density, $k_{ij}^{\text{max}} \times \frac{c_{ij}}{t_{ij}} \times \Delta \tau$. Intersections present additional limitations on flow. Link flow is therefore expanded to $q_{ij}^\tau$ for time interval $\tau$. Cell variables are assumed to change only with time intervals.

This work used the Visual Interactive System for Transport Algorithms (VISTA) (Ziliaskopoulos and Waller, 2000), a SBDTA software that uses CTM for flow propagation. VISTA takes the additional step of discretizing and tracking individual vehicles. Therefore in VISTA, $q_{ij}^\tau \times \Delta \tau \in \mathbb{Z}$. However, it results in additional complexity in finding an UE solution. VISTA separates trips and travel times by assignment intervals (typically 15min in duration) – denote by $T$ the set of assignment intervals (ASTs). Simulation time was extended to allow all vehicles to exit the network. $v_{od}^t$ is the trips from origin $o$ to destination $d$ departing during some assignment interval $t \in T$. ($v_{od} = \sum_{t \in T} v_{od}^t$). For the purposes of finding time-dependent shortest paths, travel times are aggregated by assignment interval as well; $c_{ij}^\tau$ is the average travel time on
The equilibrium condition (1.1) is then rewritten as
\[
\forall_{o,d\in Z} \forall_{t\in T} \forall_{\pi \in \Pi_{od}} (c^t_\pi = \min_{\pi' \in \Pi_{od}} (c^t_{\pi'}) \lor f^t_\pi = 0)
\] (1.5)

The gap in (1.2) becomes
\[
\delta^t_{od} = \sum_{\pi \in \Pi_{od}} \left(f^t_\pi \left(c^t_\pi - \min_{\pi' \in \Pi_{od}} (c^t_{\pi'})\right)\right) 
\] (1.6a)
\[
\delta = \sum_{o,d\in Z} \sum_{t\in T} \delta^t_{od} 
\] (1.6b)

For VISTA it is also meaningful to refer to excess cost for individual vehicles, which experience different travel times depending on flow propagation. For some vehicle \(a \in v^t_{od}\) let \(c_a\) be its travel time, and \(\delta_a\) be its excess cost. Then
\[
\delta_a = c_a - \min_{\pi' \in \Pi_{od}} (c^t_{\pi'}) 
\] (1.7)
\[
\delta\text{ can also be defined in terms of vehicle gap, specifically}
\]
\[
\delta = \sum_{o,d\in Z} \sum_{t\in T} \sum_{a \in v^t_{od}} \delta_a 
\] (1.8)

Equilibrium algorithms often follow three general steps, shown in Figure 1.1:

1. Calculate link travel times
2. Find the shortest path per OD (for STA) or ODT (for DTA)
3. Re-distribute some fraction of trips to the shortest path

**Figure 1.1** Equilibrium Algorithms
Step 3, redistributing trips, is the most challenging since naïve algorithms will result in slow convergence to the UE. The fastest STA algorithms take advantage of its typically convex, differentiable link travel time functions to choose optimal search directions. However, the complexity of incorporating congestion propagation restrictions in DTA prevents the use of such analytical algorithms. CTM, in particular, is in general not differentiable due to its discretization. As a result, DTA software often utilize less efficient STA algorithms such as the Method of Successive Averages (MSA) (Sheffi, 1984), which redistributes a standard step-size proportion of trips to the shortest path each iteration. Formally, for STA, let $\pi_{od}(n)$ be the shortest path found for OD pair $od$ at iteration $n$ – typically one path per OD (in STA) or origin-destination-assignment interval (ODT) (in VISTA) is found per iteration. Let $f_{\pi}(n)$ be the flow assigned to path $\pi$ at iteration $n$, and define $\Pi_{od}(n) = \{\pi_{od}(n')|n' \leq n\}$ is the subset of $\Pi_{od}$ including those shortest paths found at any iteration up to and including $n$. Define $\lambda_{od}(n) = \frac{1}{n}$ to be the step size. Then trip re-distribution for MSA is defined as follows in Figure 1.2:

For all $o,d \in Z$

$$f_{\pi_{od}(n)}(n) \leftarrow \lambda_{od}(n) \times v_{od}$$
For all $n' < n$

$$f_{\pi_{od}(n')}(n) \leftarrow (1 - \lambda_{od}(n)) \times f_{\pi_{od}(n')}(n - 1)$$

**Figure 1.2** Trip redistribution for the Method of Successive Averages
VISTA uses the same definition, except varying with assignment intervals $t \in T$ to use $\lambda_{od}(n), v_{od}(n), F_{od}(n), \pi_{od}(n))$. Varying step sizes per OD and to minimize $\delta$ with each iteration, as used in the well-known Frank-Wolfe algorithm (FW) (Frank and Wolfe, 1956), can also be applied to DTA. However, SBDTA requires significant additional computation time because it uses simulation calculate link travel times. With the sub-optimal step-sizes used in MSA or FW-like algorithms, in DTA convergence to a $\delta$ of a few percent on large city networks can require days. Remark: in STA, MSA and FW typically defined without the notion of path flows by taking the $\lambda_{od}(n)$ weighted average of the resulting link flows for the all-or-nothing assignment to $\pi_{od}(n)$ with the link flows from the previous iteration. Formally, the all-or-nothing assignment for paths $\pi_{od}(n)$ is the link flows

$$q_{ij} = \sum_{o,d \in \mathbb{Z}} \left\{ v_{od}(n) \begin{cases} \text{if } [i,j] \in \pi_{od}(n) \\ 0 \text{ else} \end{cases} \right\}$$

However, incorporating paths is required for tracking individual vehicles in VISTA and necessary for certain techniques in this work.

In VISTA, the default method of determining the traffic assignment is based on Simplicial Decomposition (SD) (Smith, 1983) and utilizes outer sequences of path generation to augment the path set and inner Dynamic User Equilibration (DUE) iterations, to move closer towards a restricted equilibrium (see Figure 1.3). A typical cycle is 5 iterations of path generation followed by 5 iterations of DUE, although user-selected variations are possible. To avoid the potentially subjective tweaking of
parameters, typical cycles were used for comparisons. Note that these solution algorithms are entirely deterministic. Multiple runs will yield the exact same results. Convergence techniques discussed in this work may yield different results only because discretized vehicles were assigned to a continuous distribution using a random number generator. Due to the large number of trips considered, the effect of random assignment is small.

1.2 Static ‘Warm starting’

Dial (2006) showed that for Algorithm B, STA computation time to reach a specified cost gap can be reduced by starting with an existing solution. Algorithms
which rely on an analytical determination of the best descent direction might similarly benefit. However, improvement in older algorithms such as FW is limited due to the slow convergence at later iterations. Because DTA is largely limited to less optimized algorithms, determining the extent of the benefit from warm-starting is one of the results of this study.

The concept was applied to DTA by attempting to use paths from an existing solution for the static model to improve the convergence rate of DTA. Such a warm start could significantly reduce computation time required for DTA convergence due to the relative ease of determining the static user equilibrium. After the computation of a static UE, transferring the solution into a warm start for DTA involves the determination of the set of feasible paths as well as the distribution of vehicles among those paths. Heuristics for creating an initial path set based on the static assignment are developed, tested and analyzed on networks of varying size.

1.3 Heuristic improvements in the Method of Successive Averages

Discrete vehicles in VISTA presents the opportunity to heuristically modify MSA to prioritize selected individual vehicles for redistribution. At iteration $n$, the basic version (based on MSA) relies on random number generators to shift vehicles to the new shortest path $\pi_{od}^t(n)$ per OD with the probability of route modification $\lambda_{od}^t(n) = \frac{1}{n}$, although the high number of trips reduces the effect of random variation. Numerical results show that MSA applied to VISTA moves the network towards UE.
The fact that MSA moves a randomly selected subset of vehicles, the size of which is determined by iteration alone, suggests that a more optimal vehicle choice accounting for the existing assignment could improve convergence. In static traffic assignment and continuous DTA models, choice in vehicle movement can be optimized by the use of gradient-based methods which determine the best descent direction of the cost gap function. The same methods cannot be applied rigorously to the discretized CTM. However, it adds the additional choice of which vehicles to be moved. Heuristics for selecting the optimal vehicles to be moved are compared with MSA and with each other. Methods considered include deterministic as opposed to random selection of the $\lambda_{od}^t(n) \times \nu_{od}^t$ flow moved per ODT in iteration $n$ (see Figure 1.4 for the

![Diagram](image)

**Figure 1.4** Method of Successive Averages-based heuristics algorithm flow
general algorithm for MSA-based heuristics in VISTA). Gradient-inspired methods of moving different proportions of vehicles per ODT were also tested. Previous work (Mahut et al., 2007; Sbayti et al., 2007; Lu et al., 2009; Tong and Wong, 2010) proposed heuristics and compared with MSA. However, this research effort compares these methods with each other and also tests new heuristics.

The remainder of the thesis is organized as follows: §2 and §3 introduce and analyze static warm-starting, and §4 and §5 do the same for MSA-based heuristics for DTA. §6 discusses conclusions.

2 Methods of Static Warm Starting

VISTA begins the search for an UE by finding paths. Typically, 5 iterations of path generation are run, providing up to 5 paths per ODT and assigning vehicles. The warm-start techniques were used in place of the initial path generation iterations of VISTA to reduce the time required to compute an initial path set as well as improve the effectiveness of later cycles of DUE and path generation. Although STA was run to provide a much more converged solution than the initial 5 path generation iterations would in DTA, the relative simplicity of STA as well as the availability of more optimal convergence methods allows an improvement in computation time while increasing the size of the initial set of viable paths. The drawback is that the static assignment cannot account for the complexity of DTA, so a good solution for STA may not be a good solution in DTA. A poor flow assignment will waste time in DTA moving vehicles to
viable paths. However, it is believed to be a better solution than starting from free-flow travel times. This work was first reported in Levin et al. (2011), with further analysis discussed in this paper.

The static model was based on commonly used Bureau of Public Roads (BPR) link cost functions:

\[ c_{ij} = \hat{c}_{ij} \left( 1 + \alpha_{ij} \left( \frac{q_{ij}}{q_{ij}^{\text{max}}} \right)^{\beta_{ij}} \right) \]  

(2.1)

with constants for calibration \( \alpha_{ij} \) and \( \beta_{ij} \) (usually taken to be 0.15 and 4, respectively). The same capacity value was used for both STA and DTA, although \( \frac{q_{ij}}{q_{ij}^{\text{max}}} > 1 \) is a possibility in STA. For heuristical variations in the initial assignment, FW was used to achieve convergence within a reasonable gap due to its ease in programming. However, more efficient algorithms could be used to solve the same model yet produce the same warm-start heuristics due to the uniqueness of link flows in the static UE (Sheffi, 1984).

2.1 Determining the initial path set

The goal of static warm start is to quickly determine an initial feasible set of paths and flow assignment. Since DUE will optimize the assignment for a given path set, the optimality of the path set is paramount. A warm start path set that contains few of the final DTA UE paths requires computational time to find those correct paths and then move vehicles. Similarly, a path set that includes paths not used in the DTA UE results in computational time spent moving vehicles off of those paths. The ideal warm start
would identify most, if not all, of the DTA UE paths used, as well as assign vehicles appropriately. On the other hand, potential causes of the warm start path set being incomplete or over-inclusive are the fact that many of the paths are the shortest in non-UE assignments, and the difference between link travel time from the BPR function and as a result of CTM restrictions on flow. The FW solution to static assignment does not record flows per path, and even if it did it would suggest fractional flows incompatible with CTM. As a result, heuristics for determining the path set and flows are developed and compared.

2.1.1 All-or-nothing paths

The All-or-Nothing (AON) paths heuristic is simply including $\Pi_{od}(n)$ for $n$ iterations of FW. The reasoning is that some flow is on each of the AON paths in the STA solution, and thus all paths might be viable in DTA. Furthermore, the large number of paths in the AON path set perhaps makes it the most likely heuristic to include all paths in the DTA UE.

The path set can quickly become overwhelmingly large, and to reduce space requirements duplicate paths were not stored. A simple hashing scheme based on OD and path size ideally makes the time for identifying duplicate paths after $n$ iterations $O\left(\frac{n}{|OD| \times \text{path sizes for OD}}\right)$. Path size was used to limit collisions. A more complex hash function based on some subset of the links in a path would further reduce collisions, but at the cost of additional computation time per path. Despite the hashing, the large path
size is still a considerable limitation for large networks in terms of space and vehicle assignment.

2.1.2 Paths within $\varepsilon$ of the shortest path

Assuming the static UE and dynamic UE are similar, the minimum cost paths in the static UE solution would be viable paths in DTA. Assigning the entire OD flow onto a single path, however, would result in greatly increased travel times due to link capacity limitations. Furthermore, because the perfect UE was not found, many used paths had travel time close to but not exactly that of the minimum cost path. To include these paths, paths within an additional $\varepsilon$ of the cost of the shortest path were found through an extension to Dijkstra’s (1959) algorithm in the hopes that vehicles on these paths would experience travel time close to that in the dynamic UE with a good trip assignment. Precisely, the warm-start path set per OD for $n$ iterations of FW was $\Pi'_{od} = \{\pi \in \Pi_{od}(n) | c_\pi \leq (1 + \varepsilon) \times \min_{\pi' \in \Pi_{od}(n)}(c_{\pi'})\}$. Paths containing cycles were ignored. This is similar to the $k$-shortest paths problem (see Dreyfus, 1969 for a discussion of some solution algorithms). Rather than iteratively finding the $k$th shortest path until some threshold is reached, a trace through the predecessor labels was used to find simple paths within an additional $\varepsilon$ of the shortest path cost.
2.1.3 Origin-Based Algorithm

Bar-gera’s (2002) Origin-Based Algorithm (OBA) efficiently computes the static UE solution in time linear with network size and creates a set of paths and flows which were used for the warm start. A C++ implementation of OBA from Bar-gera’s website was used (http://www.bgu.ac.il/~bargeratntp/).

2.2 Assignment of trips

A distribution of trips close to the UE results in less computation time required after the warm-start. The complexity of the distribution is compounded by the fact that the conversion from OD trips in static to ODT trips in DTA creates the possibility of different flow proportions for each time period. For time periods with a higher proportion of trips, link capacity is a considerable factor when assigning flow. Since flow assignment to paths in STA is not unique, determining likely path flows relied on testing several heuristics maximize the effectiveness of the warm start.

Each path in the subset of the path set for each OD was assigned a weight based on the heuristics, and vehicles were assigned according to the proportion of weights using a random number generator. Although repeated warm-starts might result in different path assignments, this method assigned an average of 20-30 vehicles to each path in the tested networks, offsetting the effects of random assignments.
2.2.1 Equal proportions

This method used a random number generator to assign $v_{od}$ trips in VISTA to each $\pi \in \Pi_{od}(n)$ from FW with probability $\frac{1}{|\Pi_{od}(n)|}$. Spreading vehicles evenly avoids the pitfall of placing the entire OD demand on a single path, although lower capacity paths may still experience unrealistic congestion. Nevertheless, this provides a base case to compare other heuristics of flow assignment against.

2.2.2 Density-based proportions

Heuristics based on the $\frac{q_{ij}}{q_{ij}^{max}}$ ratio for $[i,j]$, also referred to as density in STA, aggregated over the links at the converged state, were used to determine path flow proportions. One heuristic referred to as direct density assigned path flow proportional to the highest density ratio per link on the path. Specifically, a random number generator assigned vehicles to path $\pi \in \Pi_{od}(n)$ with probability

$$\frac{\max_{[i,j] \in \pi} \left( \frac{q_{ij}}{q_{ij}^{max}} \right)}{\sum_{\pi' \in \Pi_{od}(n)} \max_{[i,j] \in \pi'} \left( \frac{q_{ij}}{q_{ij}^{max}} \right)}$$

per OD. This assumes that the higher density is a result of a higher saturation of the path in the STA solution, and correspondingly assigns more vehicles to the path in DTA. The opposite heuristic, which assigned path flow inversely proportional to the highest density ratio per link with proportion
\[
\frac{1}{\max_{|i,j| \in \Phi} \left( \frac{q_{ij}}{q_{ij}^{\text{max}}} \right)} \quad \text{was inspired by the theory that drivers using the static UE as a representation of traffic would seek to avoid congestion.}
\]

2.2.3 Origin-Based Algorithm

OBA enumerates paths and flows, and proportional flows of discretized vehicles were assigned to paths in DTA through random selection. This differs from FW based heuristics in that path flows generated from the STA solution algorithm, rather than heuristical guesses, are used for flow assignment. It is reasonable to believe that these might be more effective flow assignments for the warm-start.

3 Analysis of Static Warm Starting

Convergence using the described methodologies as a warm-start was tested in VISTA against each other and VISTA’s default. Results indicate that OBA provided a good warm start on the tested networks, and furthermore that heuristics from Frank-Wolfe may reduce computation time despite resulting in a slightly worse convergence than VISTA’s default.

3.1 Test networks

Methodologies were tested on two networks of different sizes: Sioux Falls and Anaheim (see Table 3.1). Sioux Falls is a well-known, relatively small network that has
been used in many publications. The specific data used for this test was previously published in LeBlanc et al. (1975). Anaheim was chosen to test the warm-start techniques on a larger network. Link properties and network demand were obtained from Bar-Gera's website. Dynamic trips were distributed according to a bell-curve inspired demand profile (see Figure 3.1). This demonstrated the effect of a static warm start on varying demand in the dynamic model.

### 3.2 Initial computational time

The computation times required to find the initial path set (path generation or warm-start) are compared in Table 3.2. FW was run to a minimum relative gap of $1.0 \times 10^{-10}$ or a maximum of 500 iterations. OBA was run to a minimum relative gap of $1.0 \times 10^{-10}$ or a maximum of 100 main iterations. FW required significantly less computation time than VISTA for Sioux Falls but scaled worse. OBA performed much better for both Sioux Falls and Anaheim, demonstrating that finding a static UE can be

<table>
<thead>
<tr>
<th>Zones</th>
<th>Nodes</th>
<th>Links</th>
<th>Trips</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sioux Falls</td>
<td>24</td>
<td>24</td>
<td>76</td>
</tr>
<tr>
<td>Anaheim</td>
<td>38</td>
<td>416</td>
<td>914</td>
</tr>
</tbody>
</table>

**Table 3.1 Test networks**

**Figure 3.1 Demand profile (weights add to 1)**
many orders of magnitude faster than finding a dynamic UE. Despite the higher computation times, Frank-Wolfe is orders of magnitude simpler for implementation by practitioners. Nevertheless, the FW based heuristics described here could also be applied to the output from OBA, since STA has an unique UE in terms of link flows (Sheffi, 1984). Computation times for later steps are not reported because later steps for each methodology involved running the same program on the same network with a different initial assignment.

### 3.3 Convergence results

Table 3.3 presents the true (initial) gap and final gap after 30 iterations of DUE for path generation and warm-start methods. Three measures are provided: the total travel time, cost gap per vehicle, and cost gap % (the cost gap as a percent of total travel time). A comparison of initial gap demonstrates that the warm-start methods, with the exception of OBA, probably do not generally outperform VISTA path generation itself. This is surprising, but may be improved through future work on the warm-start. Nevertheless, the numerical results showing that all warm-start methods, on both networks, produce an initial cost gap % that is similar to that of VISTA path generation.

<table>
<thead>
<tr>
<th></th>
<th>Time (s)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sioux Falls</td>
<td>Anaheim</td>
</tr>
<tr>
<td>VISTA Path Generation</td>
<td>32</td>
<td>97</td>
</tr>
<tr>
<td>Origin-based Algorithm</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Frank-Wolfe</td>
<td>3.7</td>
<td>87</td>
</tr>
</tbody>
</table>

**Table 3.2 Computation time for finding paths**
Table 3.4 Warm-start on Sioux Falls and Anaheim

<table>
<thead>
<tr>
<th></th>
<th>VISTA</th>
<th>OBA</th>
<th>All-or-nothing paths</th>
<th>Paths within 5% of the shortest</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Equal proportions</td>
<td>Direct density</td>
<td>Inverse density</td>
<td></td>
</tr>
<tr>
<td><strong>Sioux Falls</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>True gap</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Travel time (hr)</td>
<td>10708.07</td>
<td>7023.5</td>
<td>8863.76</td>
<td>10086.68</td>
<td>10288.11</td>
<td>10365.83</td>
<td></td>
</tr>
<tr>
<td>Cost gap per vehicle (s)</td>
<td>82.68</td>
<td>32.96</td>
<td>60.74</td>
<td>104.62</td>
<td>98.88</td>
<td>111.75</td>
<td></td>
</tr>
<tr>
<td>Cost gap %</td>
<td>0.62</td>
<td>0.38</td>
<td>0.55</td>
<td>0.83</td>
<td>0.77</td>
<td>0.86</td>
<td></td>
</tr>
<tr>
<td><strong>Gap after 30 DUE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Travel time (hr)</td>
<td>10112.78</td>
<td>7023.5</td>
<td>9492.49</td>
<td>10332.31</td>
<td>10738.01</td>
<td>10416.85</td>
<td></td>
</tr>
<tr>
<td>Cost gap per vehicle (s)</td>
<td>46.43</td>
<td>32.96</td>
<td>51.93</td>
<td>52.19</td>
<td>54.92</td>
<td>62.35</td>
<td></td>
</tr>
<tr>
<td>Cost gap %</td>
<td>0.37</td>
<td>0.38</td>
<td>0.44</td>
<td>0.40</td>
<td>0.41</td>
<td>0.48</td>
<td></td>
</tr>
<tr>
<td><strong>Anaheim</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>True gap</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Travel time (hr)</td>
<td>49890.45</td>
<td>50312</td>
<td>45598.55</td>
<td>142298.68</td>
<td>142687.8</td>
<td>144177.8</td>
<td></td>
</tr>
<tr>
<td>Cost gap per vehicle (s)</td>
<td>611.18</td>
<td>431</td>
<td>707.84</td>
<td>1871.2</td>
<td>1878.74</td>
<td>1902.6</td>
<td></td>
</tr>
<tr>
<td>Cost gap %</td>
<td>17.02</td>
<td>11.90</td>
<td>21.57</td>
<td>18.27</td>
<td>18.29</td>
<td>18.33</td>
<td></td>
</tr>
<tr>
<td><strong>Gap after 30 DUE</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Travel time (hr)</td>
<td>41747.21</td>
<td>43567</td>
<td>39910.14</td>
<td>46154.07</td>
<td>43292.83</td>
<td>44549.08</td>
<td></td>
</tr>
<tr>
<td>Cost gap per vehicle (s)</td>
<td>244.47</td>
<td>159</td>
<td>339.9</td>
<td>363.07</td>
<td>232.9</td>
<td>329.31</td>
<td></td>
</tr>
<tr>
<td>Cost gap %</td>
<td>8.14</td>
<td>5.07</td>
<td>11.83</td>
<td>10.93</td>
<td>7.47</td>
<td>10.27</td>
<td></td>
</tr>
</tbody>
</table>

indicates that warm-starts may have the potential to perform the function of path generation in DTA at reduced computation time.

The significant difference in total travel time for the initial gap on the paths within 5% of the shortest method applied to Anaheim suggests that the initial state may be a local minimum in cost gap or even leading towards a different UE than the one found starting with VISTA path generation. In DTA Nie (2010) demonstrated that multiple equilibria are possible. Nevertheless, the greatly different total travel time and cost gap are not reflected in the final gap, for which the direct density variation initial
assignment performs better than VISTA. However, proportional results are not found in Sioux Falls. In both networks, the equal proportions method results in a final gap similar to that of VISTA, indicating that the success of flow density heuristics depends on network properties. Specifically, the longer paths and correspondingly higher travel times observed in Anaheim may favor alternate paths than those in the static UE due to more vehicles from earlier periods still on the network during the peak periods.

The all-or-nothing paths technique consistently produced a lower total travel time but a higher cost gap. Since all-or-nothing adds many more paths, it may result in lower congestion as vehicles are spread out more but there is a higher probability of use of non-optimal paths. It is worth noting here that the cost gap is measured from the shortest path available per ODT. As a result, the larger number of paths might result in an unfair comparison in terms of cost gap. Nevertheless, all-or-nothing paths converged to a gap not excessively higher than VISTA and only slightly higher than the best paths within $\varepsilon$ of the shortest method.

The origin-based algorithm produced excellent results considering the minimal computation time required for the initial path set (see Table 3.3), having around a 30% lower cost gap per vehicle for both networks. The fact that OBA produced a higher cost gap % than VISTA in Sioux Falls can be explained by the decreased total travel time. The enumeration of paths and flows from OBA appears to have resulted in a warm-start that improves the convergence of the dynamic model. Tests on other networks would be necessary to confirm this result in general, but the significant good results on two
networks is promising. Furthermore, this offers additional evidence that static traffic assignment can be effectively used as a warm-start. Even if the cost gap were no better than that produced by VISTA, the greatly reduced computation time would be of benefit. The results merit further investigation into the similarities between the static and dynamic UE and how OBA – or other warm-start techniques – take advantage of those similarities.

3.4 Path usage

To attempt to explain why certain methods were more effective than others, the number of initial paths used in the final solution for Sioux Falls was analyzed. Computing the true gap led to the addition of up to one extra path per ODT. That could add a significant number of additional paths which might be used in the final solution after 30 DUE. Also, movement of vehicles during the 30 iterations of DUE could result in fewer used paths, as observed in the OBA results (see Table 3.4). It was hypothesized

<table>
<thead>
<tr>
<th>Path usage</th>
<th>Initial paths</th>
<th>Final used paths</th>
<th>Percentage of initial paths used at convergence</th>
</tr>
</thead>
<tbody>
<tr>
<td>VISTA path generation</td>
<td>528</td>
<td>989</td>
<td>46.61</td>
</tr>
<tr>
<td>Origin-based Algorithm</td>
<td>607</td>
<td>553</td>
<td>9.76</td>
</tr>
<tr>
<td>All-or-Nothing</td>
<td>1423</td>
<td>1059</td>
<td>34.37</td>
</tr>
<tr>
<td>Paths within 5% of the shortest</td>
<td>Equal proportion</td>
<td>803</td>
<td>1023</td>
</tr>
<tr>
<td></td>
<td>Direct density</td>
<td>803</td>
<td>962</td>
</tr>
<tr>
<td></td>
<td>Inverse density</td>
<td>803</td>
<td>1005</td>
</tr>
</tbody>
</table>
that the best warm-start would have the most paths used at the final solution. However, OBA definitively produced the best warm-start results, yet had the least percentage of initial paths used – and the least number of final paths used.

However, a plot of the % of initial paths used in the final solution against the cost gap % does not appear to have a pattern (see Figure 3.2). Therefore the addition and use of more paths from the warm-start cannot be definitively labeled as an advantage or disadvantage. As demonstrated by the lower cost gap and much lower total number of paths used from the OBA warm-start, the initial assignment must be showing potential to induce the addition of better paths during the computation of the initial gap. The results from OBA question the intuition that the best warm-start provides paths and flows most similar to the UE.

3.5 Artificial increase in congestion in the static User Equilibrium

\( v_{od} \) was scaled in the static model (VISTA demand remained constant) to test whether that improved the initial assignment for the warm-start. Table 3.5 shows
surprising results: on Sioux Falls, all heuristics using paths within 5% of the shortest resulted in little change for 150% demand, but a significant decrease in cost gap – both cost gap % and per vehicle – for 200% demand. Each heuristic also shows an increase in the true gap for the 200% demand in static. Additional demand encourages the use of longer paths, suggesting that the dynamic UE benefited from the addition of those longer paths. The soft link capacity of the static model likely underestimates the hard capacity limitations enforced by CTM. (Note that a network-wide increase in demand is equivalent to a network-wide decrease in link capacity for static). Whether a demand

Table 3.5 Warm-start on Sioux Falls: varying congestion in static

<table>
<thead>
<tr>
<th></th>
<th>Sioux Falls</th>
<th>VISTA</th>
<th>Paths within 5% of the shortest</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Equal proportions</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>100% demand</td>
<td>150% demand</td>
</tr>
<tr>
<td>True gap</td>
<td>Travel time (hr)</td>
<td>10708.07</td>
<td>10086.68</td>
<td>9880.5</td>
</tr>
<tr>
<td></td>
<td>Cost gap per vehicle (s)</td>
<td>82.68</td>
<td>104.62</td>
<td>94.17</td>
</tr>
<tr>
<td></td>
<td>Cost gap %</td>
<td>0.62</td>
<td>0.83</td>
<td>0.76</td>
</tr>
<tr>
<td>Gap after 30 DUE</td>
<td>Travel time (hr)</td>
<td>10112.78</td>
<td>10332.31</td>
<td>10146.41</td>
</tr>
<tr>
<td></td>
<td>Cost gap per vehicle (s)</td>
<td>46.43</td>
<td>52.19</td>
<td>51.13</td>
</tr>
<tr>
<td></td>
<td>Cost gap %</td>
<td>0.37</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Direct density</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>100% demand</td>
<td>150% demand</td>
<td>200% demand</td>
</tr>
<tr>
<td>True gap</td>
<td>Travel time (hr)</td>
<td>10708.07</td>
<td>10288.11</td>
<td>10742.09</td>
</tr>
<tr>
<td></td>
<td>Cost gap per vehicle (s)</td>
<td>82.68</td>
<td>98.88</td>
<td>104.99</td>
</tr>
<tr>
<td></td>
<td>Cost gap %</td>
<td>0.62</td>
<td>0.77</td>
<td>0.78</td>
</tr>
<tr>
<td>Gap after 30 DUE</td>
<td>Travel time (hr)</td>
<td>10112.78</td>
<td>10738.01</td>
<td>10654.84</td>
</tr>
<tr>
<td></td>
<td>Cost gap per vehicle (s)</td>
<td>46.43</td>
<td>54.92</td>
<td>59.47</td>
</tr>
<tr>
<td></td>
<td>Cost gap %</td>
<td>0.37</td>
<td>0.41</td>
<td>0.45</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inverse density</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>100% demand</td>
<td>150% demand</td>
<td>200% demand</td>
</tr>
<tr>
<td>True gap</td>
<td>Travel time (hr)</td>
<td>10708.07</td>
<td>10365.83</td>
<td>10073.85</td>
</tr>
<tr>
<td></td>
<td>Cost gap per vehicle (s)</td>
<td>82.68</td>
<td>111.75</td>
<td>104.73</td>
</tr>
<tr>
<td></td>
<td>Cost gap %</td>
<td>0.62</td>
<td>0.86</td>
<td>0.83</td>
</tr>
<tr>
<td>Gap after 30 DUE</td>
<td>Travel time (hr)</td>
<td>10112.78</td>
<td>10416.85</td>
<td>10565</td>
</tr>
<tr>
<td></td>
<td>Cost gap per vehicle (s)</td>
<td>46.43</td>
<td>62.35</td>
<td>62.09</td>
</tr>
<tr>
<td></td>
<td>Cost gap %</td>
<td>0.37</td>
<td>0.48</td>
<td>0.47</td>
</tr>
</tbody>
</table>
increase in static for other networks might produce similar improvements remains to be seen.

3.6 Summary of warm-start results

An analysis of various warm-start techniques and heuristics implies that static warm-starting has the potential to improve both computation times and possibly rate of convergence with respect to iterations of CTM based DTA. The overview of heuristics warrants further in-depth study on more networks to better explain the effectiveness of the tested heuristics.

4 Methodology of Method of Successive Averages-based heuristics

MSA-based heuristics attempt to select the optimal vehicles to move for improved convergence, and were compared against the basic MSA of moving random vehicles. Intuitively some re-assignment of vehicles must be the best direction for moving towards the UE, although calculating that direction analytically is difficult if not impossible in simulation-based CTM. Instead, a variety of heuristics were compared in an attempt to come close to the best direction, some introduced in other work. This comparison was first reported in Pool et al. (2012), which tested and analyzed these methods on the Austin downtown, Williamson County, and Austin regional networks, and for which this author was the sole programmer. For this paper these methods were
tested on Sioux Falls and Anaheim to be more comparable to results from the static warm-start.

Except for methods which selected specific vehicles, vehicle re-assignment per \( odt \) and iteration \( n \) was random with probability \( \lambda_{od}^t(n) \). One side effect is that if the latest shortest path already exists in the path set, some selected vehicles may already be on the shortest path. This effect was observed in the comparison of methods across Sioux Falls and Anaheim (§5.3).

4.1 Partial demand loading

MSA initially distributes all vehicles onto a single path per ODT, resulting in unnatural travel times until vehicles are spread over a larger set of paths. Florian et al. (2008) suggest partial demand loading for the initial assignment, spreading out vehicles over \( m \) shortest paths. For iteration \( m \) of partial demand loading, the shortest path per ODT is found and \( \frac{1}{m} \times v_{od}^t \) is assigned to it. Simulation determines the travel time on that path, allowing new paths to be found the next iteration. This spreads out vehicles from the first iteration, preventing the early iterations of MSA from being characterized by a high cost gap due to few available paths.
4.2 Re-assignment of vehicles from only the least converged Origin-Destination-Assignment Intervals

In many observed assignments with a relatively small cost gap, a small number of ODTs contribute disproportionately to the total cost gap. These worst ODTs often have a correspondingly disproportionate number of trips, so spreading out the vehicles is likely beneficial to convergence. Other ODTs that are relatively converged, in that a change of path will result in little reduction to the cost gap, might actually have vehicles moved to a path that becomes significantly worse after the spreading out of other trips. The ODT sort heuristic attempts to avoid that by moving \( \lambda_{a_d}^t(n) = \frac{1}{n} \) vehicles for only the \( \rho \% \) of ODTs that contribute most to the gap. Formally, ODTs were sorted by \( \delta_{a_d}^t \) descending, and vehicles were moved for the top \( \rho \times |OD|^2|T| \). The optimal cutoff point is unknown, so three values of \( \rho \) were tested: 25\%, 50\%, and 75\%.

4.3 Selection of the worst vehicles

MSA randomly chooses vehicles to move, which may move vehicles experiencing a small excess cost. These heuristics move the worst vehicles first in an attempt to improve convergence. The initial definition of this idea calculates \( \delta_a \) for each vehicle \( a \in v_{a_d}^t \) and sorts in descending order, per ODT. The first \( \lambda_{a_d}^t(n) \times v_{a_d}^t \) vehicles per ODT are moved to the shortest path (ODT vehsort). A second version prioritizes vehicle movement based on the cost of the path, again per ODT (ODT pathsort). Simulation results in the possibility of distinct vehicles on the same path experiencing different
travel times. Individual vehicle cost gap can result from time-dependent traffic congestion, whereas the path cost takes into account average travel times along the path.

A third variation was tested which sorts vehicles by $\delta_a$ not per ODT (vehsort). The first $\lambda_{od}^t(n) \times \sum_{o,d\in Z} \sum_{t\in T} v_{od}^t$ vehicles were moved to the new shortest path for their ODT. This is a further extension of the idea that moving vehicles in well-converged ODTs might be less productive than a focus on the worst ODTs, which account for much of the cost gap. However, this differs from ODTsort by targeting individual vehicles rather than the worst ODTs. High-gap vehicles from otherwise well-converged ODTs can still be re-assigned.

4.4 Time-varying step size

Mahut et al. (2007) suggested that later assignment intervals cannot be truly converged until previous assignment intervals are stabilized. A cascade pattern with higher values at later time intervals, such as the one in Table 4.1, moves more vehicles for later ODTs in accordance with the theory. Step size is initially constant with respect to time period, then at some iteration $n_0$ begins to gradually shift into a cascade pattern. The shift finalizes into a pattern described by the reset parameter $\Delta\lambda$, which is the iteration difference in the lambda value between period $t$ and $t + 1$. In practice, the optimal parameters are unknown, and effectiveness is likely to vary with respect to the combination of cascade parameters and network.
4.5 Gradient based heuristics

Continuous models benefit from algebraic definitions allowing the computation or close approximation of the gradient of the cost gap to determine the direction of fastest decrease. Lu et al. (2009) and Chiu and Bustillos (2009) proposed using equivalent variables from the simulation, such as the level of convergence or cost gap, to select a better lambda per ODT. The average ODT gap and total ODT gap heuristics vary lambda based on the gap per ODT relative to the gap per network. Specifically, for total-gap

$$\lambda_{od}^{t}(n) = \frac{1}{n} \times \min \left( 1.5, \frac{\delta_{od}^{t}}{\sum_{o,de} \sum_{t'\in T} \delta_{od}^{t'}} \right) \quad (4.1)$$

and for avg-gap

$$\lambda_{od}^{t}(n) = \frac{1}{n} \times \min \left( 1.5, \frac{\delta_{od}^{t}/v_{od}^{t}}{\sum_{o,de} \sum_{t'\in T} \delta_{od}^{t}/v_{od}^{t}} \right) \quad (4.2)$$

These move more vehicles for ODTs which have a higher gap, but still move some vehicles for each ODT.
Two further heuristics relying on the difference in costs between the shortest paths and others. Inspired by Lu et al. (2009), relative path cost applies a path-based step-size

\[
\lambda^t(n) = \min \left( 1.5 \frac{1}{n}, \frac{c_n - \min_{\pi' \in \Pi_{od}^t} (c_{\pi'})}{\min_{\pi' \in \Pi_{od}^t} (c_{\pi'})} \right)
\] (4.3)

for each path \( \pi \in \Pi_{od}^t \) per ODT. This method is similar to a static assignment gradient-projection method by Jayakrishnan et al. (1994) without the incorporation of the second derivatives (the derivative of link travel times). Further work using an approximation of the second derivatives may be beneficial, but was not included here due to the added complexity in CTM. Relative gap sum similarly applies a step-size per ODT of

\[
\lambda_{od}^t(n) = \min \left( 1.5 \frac{1}{n}, \sum_{\pi \in \Pi_{od}^t} \frac{c_n - \min_{\pi' \in \Pi_{od}^t} (c_{\pi'})}{\min_{\pi' \in \Pi_{od}^t} (c_{\pi'})} \right)
\] (4.4)

This aggregates the vehicle movement per ODT rather than being path-specific. Due to the variation in vehicle travel times even on the same path in DTA, this could be more effective. Note that variations on lambda in §4.4 and §4.5 can be applied orthogonally to methods of selecting vehicles in §4.2 and §4.3.

4.6 Simplicial Decomposition

As discussed in §1.1, VISTA typically uses ‘outer’ cycles of finding paths combined with ‘inner cycles’ of equilibration in a manner similar to Simplicial Decomposition (SD). Smith (1983) presented an algorithm that adds paths in outer iterations and in inner iterations finds a restricted equilibrium by modifying either the assignment or step size...
depending on the distance from equilibrium for a given subset of all paths. The main difference in VISTA is that the grouping of outer iterations often adds multiple paths per ODT. Finding a restricted equilibrium for each outer iteration of path generation will greatly increase the computational time, possibly without a proportional increase in the quality of the restricted equilibrium. The addition of new shortest paths has been observed to significantly alter the relative gap.

Although one stopping criterion is having the new shortest path per ODT already in the path set, the large number of potential paths combined with the high congestion on high demand ODTs makes achieving this stopping criterion unlikely for large networks in a reasonable number of iterations. In terms of space requirements, the expressed concern by Smith (1983) of path enumeration is obviated as VISTA already stores all paths as well as discrete vehicle assignments. For general DTA applications, the significant increases in computer memory, storage, and processing time have made path enumeration much more viable even for large networks.

Due to the non-continuous nature of CTM, the same proof of convergence cannot be applied here. DUE in VISTA tests multiple step sizes, choosing the one resulting in the lowest cost gap. In this work, step sizes per ODT and vehicle movement were based on the above heuristics (i.e. $\lambda_{od}^t(n) = \frac{1}{n}$, with $n$ incrementing for both path generation and equilibration). This is reasonable because as the size of the path set increases, the change in the restricted equilibrium after adding an additional path per ODT should decrease. MSA was chosen as a control comparison, although the above
methods are all compatible with (actually orthogonal to) the SD partition of iterations into outer iterations of path generation and inner iterations of equilibration. Conceivably step sizes based on some function of flows and/or cost gap (such as in one of the above heuristics) as used in Smith (1983)’s work, could improve effectiveness of equilibration. Using different functions for step size in the outer and inner partitions, as well as evaluating the effect of heuristic vehicle selection, are additional potential directions for further research.

SD is intended to improve the effectiveness of the outer iterations of path generation. One major question is how many iterations of path generation followed by how many iterations of equilibration. It is generally expected not to reach a true restricted equilibrium in a reasonable number of iterations of equilibration. Investing computation time into the inner cycles has an associated opportunity cost of not finding new paths, and equilibrating flows on a limited subset of paths is of limited benefit depending on the number and quality of paths considered. However, although not analytically proven it is believed that equilibrating improves the quality of paths found by later iterations of path generation by increasing stability of link costs.

5 Analysis of Method of Successive Averages-based heuristics

Each heuristic was tested with 30 iterations of path generation and 70 iterations of equilibration. (SD had 5 iterations of path generation followed by 5 iterations of equilibration up to 60 iterations, then 40 iterations of equilibration – a total of 30
iterations of path generation and 70 iterations of equilibration). Convergence could have been further improved, but the number of iterations was fixed in the interests of limiting computation time. Equilibration did not try several different step sizes, as it does in VISTA’s DUE, but continued with the step size and assignment heuristic from path generation using the shortest path per ODT from the existing path set. The true gap was computed by finding the shortest paths per ODT (without moving vehicles). This section, organized much like §4, compares the results of different categories of heuristics to MSA. Trends are discussed and explanations suggested. The final true gap for each method (applied after 5 iterations of partial-demand loading) is compared in

### Table 5.1 True gap comparison of all MSA-based heuristics

<table>
<thead>
<tr>
<th>Method</th>
<th>Total travel time (hr)</th>
<th>Cost gap per vehicle (s)</th>
<th>Cost gap %</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sioux Falls</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSA</td>
<td>9786.85</td>
<td>6.31</td>
<td>0.52</td>
</tr>
<tr>
<td>ODT sort</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20%</td>
<td>10783.59</td>
<td>123.47</td>
<td>9.18</td>
</tr>
<tr>
<td>50%</td>
<td>11472.41</td>
<td>113.69</td>
<td>7.94</td>
</tr>
<tr>
<td>75%</td>
<td>11247.59</td>
<td>51.31</td>
<td>3.66</td>
</tr>
<tr>
<td>ODT vehsort</td>
<td>8971.53</td>
<td>7.27</td>
<td>0.65</td>
</tr>
<tr>
<td>ODT pathsort</td>
<td>9648.95</td>
<td>4.69</td>
<td>0.39</td>
</tr>
<tr>
<td>vehsort</td>
<td>10864.68</td>
<td>54.56</td>
<td>4.02</td>
</tr>
<tr>
<td>Varying step size</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avg. ODT gap</td>
<td>8967.67</td>
<td>7.81</td>
<td>0.70</td>
</tr>
<tr>
<td>Total ODT gap</td>
<td>8954.72</td>
<td>8.38</td>
<td>0.75</td>
</tr>
<tr>
<td>Time-varying</td>
<td>9793.23</td>
<td>4.80</td>
<td>0.39</td>
</tr>
<tr>
<td>Relative path cost</td>
<td>9700.06</td>
<td>4.82</td>
<td>0.40</td>
</tr>
<tr>
<td>Relative gap sum</td>
<td>9984.91</td>
<td>7.31</td>
<td>0.59</td>
</tr>
<tr>
<td>Simplicial decomposition</td>
<td>6624.94</td>
<td>2.76</td>
<td>0.12</td>
</tr>
<tr>
<td><strong>Anaheim</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MSA</td>
<td>131991</td>
<td>411.86</td>
<td>9.07</td>
</tr>
<tr>
<td>ODT sort</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20%</td>
<td>155463.9</td>
<td>1200.54</td>
<td>22.47</td>
</tr>
<tr>
<td>50%</td>
<td>155537</td>
<td>1054.443</td>
<td>19.78</td>
</tr>
<tr>
<td>75%</td>
<td>134099.6</td>
<td>658.49</td>
<td>14.49</td>
</tr>
<tr>
<td>ODT vehsort</td>
<td>130833.5</td>
<td>355.64</td>
<td>7.91</td>
</tr>
<tr>
<td>ODT pathsort</td>
<td>129396.6</td>
<td>385.85</td>
<td>8.67</td>
</tr>
<tr>
<td>vehsort</td>
<td>138725.4</td>
<td>842.25</td>
<td>17.66</td>
</tr>
<tr>
<td>Varying step size</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avg. ODT gap</td>
<td>130743</td>
<td>361.66</td>
<td>8.04</td>
</tr>
<tr>
<td>Total ODT gap</td>
<td>133247.5</td>
<td>461.72</td>
<td>10.08</td>
</tr>
<tr>
<td>Time-varying</td>
<td>132773</td>
<td>432.60</td>
<td>9.48</td>
</tr>
<tr>
<td>Relative path cost</td>
<td>135388.4</td>
<td>530.48</td>
<td>11.39</td>
</tr>
<tr>
<td>Relative gap sum</td>
<td>14332</td>
<td>646.84</td>
<td>13.13</td>
</tr>
<tr>
<td>Simplicial decomposition</td>
<td>131067.7</td>
<td>423.30</td>
<td>9.39</td>
</tr>
</tbody>
</table>
Table 5.1, in the order they are presented above. However, due to occasional spikes in cost gap, the relative convergence rate as compared in the graphs below may be a more meaningful comparison.

### 5.1 Partial demand loading

5 iterations of partial-demand loading followed by MSA, was compared to MSA alone. Partial-demand loading showed only modest improvements on Sioux Falls and Anaheim. However, on large networks such as the Austin regional network (12478 nodes, 26998 links, 1229511 trips), partial-demand loading was observed to prevent gridlock in early iterations resulting in large numbers of vehicles not exiting. This makes link travel times from early iterations less accurate, possibly resulting in sub-optimal paths added for those iterations. This increases total computation time required to achieve a given level of convergence. Figure 5.2 compares the true gap with and without partial-demand loading. All further heuristics were tested after 5 iterations of partial-demand.

### Table 5.2 True gap comparison of partial demand loading

<table>
<thead>
<tr>
<th></th>
<th>Sioux Falls</th>
<th></th>
<th>Anaheim</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Total travel time (hr)</td>
<td>Cost gap per vehicle (s)</td>
<td>Cost gap %</td>
<td>Total travel time (hr)</td>
</tr>
<tr>
<td>MSA alone</td>
<td>9980.65</td>
<td>7.70</td>
<td>0.62</td>
<td>136430.2</td>
</tr>
<tr>
<td>MSA with 5 iterations of partial-demand loading</td>
<td>9786.85</td>
<td>6.31</td>
<td>0.52</td>
<td>131991</td>
</tr>
</tbody>
</table>
5.2 Re-assignment of vehicles from the least converged ODTs only

As shown in Table 5.1 and Figure 5.1, the ODT sort heuristics performed worse with decreases in $\rho$. This indicates that the reduced focus on only the worst gap ODTs is detrimental to the convergence rate. However, the reduced step size might be a contributing factor. $\rho \times \lambda_{od}^T(n)$ is significantly less than $\lambda_{od}^T(n)$, and the reduced number of vehicles moved might have caused the low convergence rate rather than the focus on the worst ODTs. Alternately, it is also possible that high demand ODTs monopolize the selection of vehicles moved, resulting in many low-trip, low-gap ODTs contributing to the gap but not being improved even at later iterations.

![Figure 5.1 Moving vehicles only from the least $x\%$ converged ODTs](image-url)
5.3 Selection of the worst vehicles

Moving the worst vehicles overall (vehsort) consistently showed a lack of convergence (see Figure 5.2). Since the highest gap is often found on ODTs with the highest demand, or near congested areas, vehicles from other ODTs are unlikely to be near the top of the worst vehicles list. Once lambda becomes small, vehicles from low-demand ODTs may no longer be moved. This might explain the stable cost gap percent after around the 8\textsuperscript{th} iteration. Since new routes found for high-demand ODTs at later iterations are often long, low-capacity routes that are shorter only because the short, high-capacity routes have a high vehicle density, this method might overwhelm the low-capacity routes by moving too many vehicles. Those vehicles would continue to have a

![Figure 5.2 Heuristics moving the worst vehicles](image)
high gap, and as the number of moved vehicles decreased with iteration the high-demand ODTs’ vehicles would continue to be prioritized. Considering the results from ODTsort, this suggests that priority selection of only the worst vehicles regardless of ODT may not be an improved search direction.

Selecting the worst vehicles per ODT shows some promise, although the best method of determining the worst vehicles is unclear. In Sioux Falls, choosing the worst vehicles by path improves convergence, whereas choosing the worst vehicles by gap isn’t clearly better than MSA. The opposite is true for Anaheim. The small number of links in Sioux Falls (76) may result in a small number of paths per ODT. Many of the vehicles with the worst gap may already be assigned to the shortest path. The number of vehicles moved per iteration from ODT vehsort in Sioux Falls, as shown in Figure 5.3, is far below the number selected to be moved by lambda. The number of vehicles moved in Anaheim is closer to the number selected, which may explain why ODT vehsort is consistently better than MSA there. In Sioux Falls, ODT vehsort uses 854

![Figure 5.3 Demand moved in Sioux Falls and Anaheim by ODT vehsort](image)

Figure 5.3 Demand moved in Sioux Falls and Anaheim by ODT vehsort
paths (1.62 per OD) while Anaheim uses 12881 (9.16 per OD), at their final solutions. ODT pathsort may not be as effective with respect to MSA in Anaheim because a larger number of paths per ODT means the worst vehicles are more likely to be spread out over several paths, and concentrating vehicle re-assignment on one or two worst paths cannot select the worst vehicles effectively.

Although ODT vehsort may not improve the overall cost gap in Sioux Falls, it has a significant effect on the gap distribution (see Figure 5.4). MSA’s random selection of vehicles to move results in a higher proportion of the gap being attributed to individual vehicles. That proportion is small but the difference is significant in Sioux Falls, and the proportion and difference are even more apparent in Anaheim. Figure 5.4 shows the proportion of gap from vehicles with a gap over 50% of their route travel time.

**Figure 5.4** Gap distribution – % of gap from vehicles with an individual cost gap over 50% of individual travel time
Unsurprisingly, ODT vehsort is consistently lower than MSA in this regard, since it moves the worst vehicles first. The difference between ODT vehsort and ODT pathsort is less pronounced in Sioux Falls, and in fact ODT pathsort appears to reduce this measure better than ODT vehsort at some iterations. This may be related to the lower number of vehicles moved by ODT vehsort especially in Sioux Falls. If the worst vehicles are already on the average best path (which is more likely to happen with few paths per ODT), selecting them first does not reduce this measure. Selecting vehicles from the worst path might move a higher proportion of high gap vehicles, as seen in Figure 5.3. Nevertheless, both methods distribute the gap more evenly than MSA despite the total cost gap. This measure is hoped to reveal vehicles from low demand ODTs that are rarely moved at lower lambda values yet contribute significantly to the excess cost. In Sioux Falls, 13.1% of ODTs have fewer than 5 vehicles, while in Anaheim just 7.0% of ODTs have fewer than 5 vehicles, a proportion that is not reflected in Figure 5.4. If that were an issue, forcing vehicles above some cutoff value of excess cost to be moved might minimize a disproportionate impact on the cost gap.

5.4 Step size variations per ODT

The time varying step size showed consistent improvement across iterations from MSA in Sioux Falls, but in Anaheim any improvement in convergence rate was unclear (see Figure 5.5a). It is possible that a different reset parameter may have yielded better results. However, given that the only change from MSA was the cascade
of lambdas for different time periods, it was expected that some improvement would have occurred. Considering that Mahut et al. (2007) achieved good results, this indicates that perhaps a significantly higher reset parameter is needed, or that a small change in the cascade pattern / reset parameter could create a disproportionate effect in the convergence rate. The results on Sioux Falls, along with Mahut et al (2007)’s work, indicate that this method may have potential if further work clarifying the optimal reset parameter as a function of the network is available.

Results are inconclusive for the total ODT gap and average ODT gap lambda methods. In Sioux Falls, they appear to have a slightly higher gap at the last iterations than MSA, although between iterations 40 and 60 a lower gap is observed. In Anaheim
a similar pattern is observed, with a lower gap between iterations 60 and 85 but both methods showing a higher gap after. Although the average ODT gap method does eventually reach a lower true gap than MSA after 100 iterations, it cannot be definitively labeled as better due to the gap at the arbitrarily chosen stopping point.

Relative path cost and relative gap sum, which rely even more on the gradient, also had different results in Sioux Falls and Anaheim (see Figure 5.5b). In Sioux Falls, relative path cost was consistently better than MSA, while relative gap sum was consistently slightly worse. In Anaheim, the results are more muddled; both relative path cost and relative gap sum perform better than MSA occasionally, but finish with a worse gap. This may, however, be related to the fact that the lambda factor used is only a part of the mathematical formulation for the gradient-projection (GP) flow assignment changes given by Jayakrishnan et al. (1994). Specifically, for \( \Pi_{od}^t(n) \) let \( \tilde{p}_{od}^t \) be the shortest path for \( odt \), satisfying \( \forall p' \in \Pi_{od}^t(n) \left[ c_{\tilde{p}_{od}^t} \leq c_{p'} \right] \). Then GP sets path flows for iteration \((n + 1)\) to be

\[
f_p^t(n + 1) = \max \left( 0, f_p^t(n) - \frac{\alpha^n}{s^n} (c_p - c_{\tilde{p}}) \right)
\]

where \( s^n \) is the sum of the derivatives of link travel time with respect to volume for links on \( p \) and \( \tilde{p} \) but not on both, and \( \alpha^n \) is a step-size modifier. This is mentioned here because \( s^n \) is not included in the MSA heuristic and its absence could result in a direction away from the gradient. In the smaller Sioux Falls network, it is possible that the number of different links between two paths is limited due to network topography,
causing the lack of $s^n$ to have a lesser impact. In Anaheim, however, that would not be the case, and many completely different paths would be available per ODT, which could explain the differences in effectiveness. An approximation of $s^n$ may yield significantly better results for the relative path cost heuristic.

**5.5 Simplicial Decomposition**

Alternating 5 iterations of path generation with 5 iterations of equilibration appeared quite effective in Sioux Falls, but not so much in Anaheim (see Figure 5.6). A related observation is the low number of additional paths in Sioux Falls during Simplicial Decomposition (SD) despite significant numbers of additional paths in
MSA (see Figure 5.7). This indicates that many of the new ODT shortest paths are already in the path set because of the equilibration for SD, resulting in a small gap. Additionally, although in Sioux Falls MSA finds a total of 15230 paths compared with SD finding 9554 paths, the final solution from MSA uses just 937 paths compared with SD’s using 969 paths. In contrast, on Anaheim MSA finds a total of 628065 paths, using 14844 while SD found a total of 609381 paths, using 14615. In Sioux Falls, most of the paths in SD were found during the partial demand loading. This suggests that SD may be finding and/or using higher quality paths than MSA in Sioux Falls – the interspersed iterations of equilibration appear to be effective. However, in Anaheim SD continues to find many new paths. The disparity may occur because Sioux Falls is too small to have
many good alternate paths, unlike Anaheim. It could also mean that Anaheim requires more than 5 iterations of DUE for the restricted equilibrium to be beneficial, and further work in SD could produce improvement similar to that observed in Sioux Falls on larger networks.

5.6 Summary of Method of Successive Averages-based heuristics results

The lack of consistent improvement indicates that none of the heuristics are currently definitively better than MSA. These results are in agreement with previous work by Tong and Wong (2010) comparing heuristics with MSA and finding little difference. Although Mahut et al. (2007) observed significant improvement in convergence, the parameters to achieve that improvement may vary per network and with the specific characteristics of the DTA model. The improved convergence on Sioux Falls, but not on Anaheim, from several of these heuristics suggests the potential for similar improvement with the right parameters. Further work defining these parameters as a function of the network may make these heuristics viable.
6 Conclusions

This thesis studies two possible methods of improving Dynamic Traffic Assignment (DTA) convergence rate and/or computation time on a Cell Transmission based model. Results from the static warm start were encouraging for both decreasing computation time and increasing convergence rate. Effectiveness of Frank-Wolfe based heuristics varied depending on the network, but the assignment from the Origin-Based Algorithm was consistently better than initial iterations of the default algorithm from VISTA. However, the explanation for these results was not fully clear. Further, certain increases in demand in static were able to produce disproportionate improvements in the DTA solution. The results suggest general benefit to be gained from warm starting this DTA model, and show the potential for further optimization of the warm start assignment.

The Method of Successive Averages (MSA) based heuristics had mixed results, which agrees with some previous work. Nevertheless, MSA demonstrates some potential to reduce the amount of human intervention necessary to reach a good solution, which itself is a benefit. If DTA requires days to find the solution, better it can find that solution without requiring frequent parameter adjustment during the convergence process.

The effectiveness of heuristics appears to differ based on network characteristics, which suggests certain heuristics could be beneficial for specific networks. In particular, selecting vehicles from the worst paths worked better with the
fewer paths per origin-destination pair in Sioux Falls, but selecting the worst vehicles yielded a better convergence rate on Anaheim. Some heuristics showed significant improvement on Sioux Falls but not on Anaheim, indicating that they may have potential for more general use with further study. Modification of the specific parameters of Simplicial Decomposition and step size variations might improve their effectiveness on Anaheim and other similar networks, although the optimal parameters are not known. Previous work finding improved convergence rates, as well as the mixed results in this study, indicates that further research would be valuable.
Appendix A1: Notation

\([i,j]\) \quad \text{link from} \ i \ \text{to} \ j

\(\alpha_{ij}\) \quad \text{Calibration constant for BPR cost function}

\(\beta_{ij}\) \quad \text{Calibration constant for BPR cost function}

\(A\) \quad \text{set of links}

\(c_{ij}\) \quad \text{travel time of } [i,j]

\(c_{ij}^t\) \quad \text{travel time of } [i,j] \text{ for some } t \in T

\(\dot{c}_{ij}\) \quad \text{free flow (unobstructed) travel time of } [i,j]

\(c_{\pi}\) \quad \text{travel time on path } \pi

\(c_{\pi}^t\) \quad \text{travel time on path } \pi \text{ for some } t \in T

\(\delta\) \quad \text{total excess cost (gap)}

\(\delta_{od}\) \quad \text{excess cost for } o, d \in Z

\(\delta_{od}^t\) \quad \text{excess cost for } o, d \in Z, \ t \in T

\(\hat{\delta}\) \quad \text{excess cost as a percent of total travel time}

\(\bar{\delta}\) \quad \text{average excess cost per vehicle}

\(\delta_a\) \quad \text{excess cost for some vehicle } a

\(f_{\pi}\) \quad \text{trips assigned to path } \pi

\(f_{\pi}(n)\) \quad \text{trips assigned to path } \pi \text{ at iteration } n

\(f_{\pi}^t(n)\) \quad \text{trips departing within some } t \in T \text{ assigned to path } \pi \text{ at iteration } n

\(G\) \quad \text{traffic network } (G = (N, A, V) )

\(k_{ij}\) \quad \text{density on } [i,j]

\(k_{ij}^{\text{max}}\) \quad \text{jam (maximum) density of } [i,j]

\(\lambda_{od}(n)\) \quad \text{step size for } o, d \in Z \text{ at iteration } n

\(\lambda_{od}^t(n)\) \quad \text{step size for } o, d \in Z \text{ for some } t \in T \text{ at iteration } n

\(\lambda_{\pi}^t\) \quad \text{step size for path } \pi \text{ for some } t \in T

\(\Delta \lambda\) \quad \text{reset parameter for time-varying step size}

\(l_{ij}\) \quad \text{length of } [i,j]

\(N\) \quad \text{set of nodes}

\(\Pi_{od}\) \quad \text{set of all paths from } o \text{ to } d \text{ with } o, d \in Z

\(\Pi_{od}(n)\) \quad \text{subset of } \Pi_{od} \text{ used at iteration } n

\(\pi_{od}(n)\) \quad \text{shortest path for } o, d \in Z \text{ at iteration } n

\(\pi_{od}^t(n)\) \quad \text{shortest path for } o, d \in Z \text{ departing within some } t \in T \text{ at iteration } n

\(q_{ij}\) \quad \text{flow on } [i,j]

\(q_{ij}^{\text{max}}\) \quad \text{capacity (maximum flow) of } [i,j]

\(\tau\) \quad \text{time interval in CTM}

\(\Delta \tau\) \quad \text{time interval width in CTM}

\(T\) \quad \text{set of assignment intervals}

\(V\) \quad \text{trips for the network}

\(v_{od}\) \quad \text{trips from } o \text{ to } d \text{ with } o, d \in Z

\(v_{od}^t\) \quad \text{trips from } o \text{ to } d \text{ with } o, d \in Z \text{ departing within some } t \in T

\(Z\) \quad \text{set of zones } (Z \subseteq N)
### Appendix A2: Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AST</td>
<td>Assignment interval</td>
</tr>
<tr>
<td>AON</td>
<td>All-or-Nothing assignment</td>
</tr>
<tr>
<td>BPR</td>
<td>Bureau of Public Roads</td>
</tr>
<tr>
<td>CTM</td>
<td>Cell Transmission Model</td>
</tr>
<tr>
<td>DTA</td>
<td>Dynamic Traffic Assignment</td>
</tr>
<tr>
<td>DUE</td>
<td>Dynamic User Equilibration</td>
</tr>
<tr>
<td>FW</td>
<td>Frank-Wolfe algorithm</td>
</tr>
<tr>
<td>LWR</td>
<td>Lighthill-Whitham-Richards model</td>
</tr>
<tr>
<td>MSA</td>
<td>Method of Successive Averages</td>
</tr>
<tr>
<td>OBA</td>
<td>Origin-Based Algorithm for STA</td>
</tr>
<tr>
<td>OD</td>
<td>Origin-Destination pair</td>
</tr>
<tr>
<td>ODT</td>
<td>Origin-Destination-Assignment interval tuple</td>
</tr>
<tr>
<td>SD</td>
<td>Simplicial Decomposition</td>
</tr>
<tr>
<td>STA</td>
<td>Static Traffic Assignment</td>
</tr>
<tr>
<td>SBDTA</td>
<td>Simulation-Based DTA</td>
</tr>
<tr>
<td>UE</td>
<td>User equilibrium</td>
</tr>
<tr>
<td>VISTA</td>
<td>Visual Interactive System for Transport Algorithm</td>
</tr>
</tbody>
</table>
References


