A Path Marginal Cost Approximation Algorithm for System Optimal Quasi-Dynamic Traffic Assignment

Abstract

This study introduces an efficient path-based System-Optimal Quasi-Dynamic Traffic Assignment (SOQDTA) framework that benefits from the computational efficiency of static traffic assignment models, yet captures the realism of traffic flow, with less complexity and computational burden, compared to dynamic traffic assignment models.

To solve the proposed SOQDTA problem, we have developed a novel Path Marginal Cost (PMC) approximation algorithm, based on a newly-proposed Quasi-Dynamic Network Loading (QDNL) procedure (Bliemer et al., 2014), that incorporates a first order node model, and thus produces realistic path travel times consistent with queuing theory, and similar to dynamic network loading models, but at a lower computational cost. The model considers capacity constrained static flows, residual vertical/point queues and no spillback.

The proposed SOQDTA model is applied to the test network of Sioux Falls and it is demonstrated that the model results in system optimal traffic flow patterns that improve total system travel times compared to the user equilibrium solution. In the case study experiment, the convergence of the algorithm is demonstrated using a relative gap function. A sensitivity analysis is performed to realize the impact of perturbation size on the solution quality, and a discussion is presented on the selection of perturbation size for general network applications.

Keywords: Quasi-Dynamic Traffic Assignment; System Optimal Traffic Assignment; Path Marginal Cost Approximation

Introduction

System optimal (SO) traffic assignment models belong to the class of transportation network modelling problems and have various applications in traffic management. These applications range from recurrent traffic management practices, such as congestion pricing, and traffic control/information systems, to non-recurrent traffic management, such as Incident Traffic Management (ITM) and evacuation scenarios. With recent advancements in information and communication technologies, vehicle automation (Autonomous Vehicles) and vehicle/infrastructure connectivity (Connected Vehicles), possibilities emerge for communicating and enforcing advanced traffic routing directions for efficient utilization of existing traffic network capacities. Therefore, the task of finding optimal traffic directions becomes more essential, as the required technologies will be available to facilitate the implementation and enforcement of optimal directions, regardless of possible complexities. The research reported in this paper studies and present a method for finding efficient traffic directions within reasonable computational times, to be applied to real-time decision making scenarios.

SO and user optimal (user equilibrium -- UE) traffic assignment problems, have been widely studied under both static and dynamic traffic flow assumptions. Despite the growing interest in the development and application of dynamic traffic assignment (DTA) models, static traffic assignment (STA) models are still widely used, specifically in strategic transportation planning, due to higher efficiency and scalability, and lower computational complexity. The computational efficiency of a traffic assignment model becomes even more crucial in real-time decision making applications such as emergency evacuations and incident management, compared to long-term strategic transportation planning and operational applications.

In traffic assignment, whether static or dynamic, the assumptions regarding the propagation of flow in the network (network loading) highly affect the model outputs. Therefore, besides computational efficiency, the ability of an assignment model in capturing the realism of traffic flow propagation plays a critical role in determining the quality of solution outputs.

In classic STA models, no link capacity constraints are presumed and the impact of high link flows are only captured through increased link travel times. Therefore, there have been many efforts in the literature to improve the precision and realism of STA models to generate more accurate traffic flow patterns and travel times, whilst taking advantage of their high computational tractability. Research along this path has led to a class of assignment models where link capacity constraints and/or residual queues are incorporated into static assignment. In the literature, capacity constrained static models with residual queues are referred to as Quasi-Dynamic Traffic Assignment (QDTA) models.

A recent study by Bliemer et al. (2014) has introduced an efficient path-based quasi-dynamic traffic assignment approach to alleviate the existing issues with the current capacity constrained static models. Their model considers static but capacity-constrained flows with residual vertical/point queues and no queue spillback. They have incorporated a first order node model in their quasi-dynamic network loading (QDNL) model, to compute the actual turn flows at nodes as well as residual point queues upstream of bottlenecks which can improve the accuracy of path travel time estimations. They used their proposed QDNL method to solve a path-based stochastic UE problem for general network settings. Such QDNL procedures, which generate reliable traffic flow patterns at a relatively low computational cost, can also be utilized to define and efficiently solve a path-based SO Quasi-Dynamic Traffic Assignment (SOQDTA) problem. A SOQDTA problem can generate

practical solutions at a lower computational cost than SO Dynamic Traffic Assignment (SODTA) problems.

As explained in the literature review section, one further advantage of a path-based SOQDTA problem is that it can be solved using the conventional and widely-studied algorithms developed for UE traffic assignment. However, using these methods to solve a path-based SOQDTA problem requires the computation of path marginal cost (PMC), which is defined as the derivative of the total system travel time with respect to the flow on each path. In STA, PMC is simply computed by taking the derivative from the total travel time function with respect to flow, however in DTA and QDTA, travel time is calculated through implicit functions and exact PMC computation is very challenging. A variety of studies have introduced algorithms to approximate PMC for *SODTA* (Ghali & Smith, 1995; Peeta & Mahmassani, 1995; Qian et al., 2012; Shen et al., 2007), however to the best of our knowledge, there exist no similar examples of PMC approximation for QDTA.

The contribution of the present study is twofold. First, we have developed a generic SOQDTA framework which embeds a state-of-the-art QDNL model and can benefit a variety of traffic management applications. It needs to be highlighted that the QDNL model considers capacity constrained static flows and residual vertical queues without queue spillback. It also does not directly model signalized intersections. Second, we have developed a PMC approximation algorithm that can efficiently solve this path-based SOQDTA problem for real-sized transportation networks, with realistic traffic flow assumptions and a low computational cost. The exactness of the proposed method is not guaranteed because the PMC values are approximate; however, the case study experiment demonstrates considerable improvement of the objective value as compared to the UE solution (do nothing scenario). For the case study experiment, we have applied the model to the medium-sized test network of Sioux Falls and demonstrated improvements in the total system travel time.

The following section elaborates on the existing literature in the context of this study. Next, the proposed methodology and algorithms are explained and lastly, the model is applied to the Sioux Falls network and the results are discussed and concluded.

Literature Review

The SOQDTA problem is founded on multiple components, including a quasi-dynamic network loading model, a first-order node model, and a path marginal cost approximation algorithm for solving the system optimal traffic assignment problem. Therefore, the literature review section covers these components.

In an effort to reduce the computational complexity of the dynamic network loading problem, Bliemer et al. (2014) introduced an efficient Quasi-Dynamic Network Loading (QDNL) model. Their model incorporates a comprehensive first order node model to properly constrain traffic flows, predict the average number of vehicles in the queue and locate queues upstream of bottlenecks. This model considers static traffic demand with residual vertical queues and no spillback, however produces traffic flow patterns and travel times similar to dynamic traffic assignment model by considering realistic supply-demand interactions. The QDNL model represents traffic flow characteristics in the network via link reduction factors, defined as the ratio of link out-flow to link demand, and uses these reduction factors to compute average path travel times consistent with queuing theory. Their model proposes a reasonable balance between static and dynamic traffic assignment, which benefits from the computational efficiency of static traffic assignment models, while sufficiently capturing the

spatial interactions of traffic flows. The present study has therefore been founded on this QDNL model, which will be explicitly explained in the methodology section.

Node models are an essential component of DNL models, and go hand-in-hand with link models to determine the propagation of flow through the nodes and on the links. Despite the importance of node models in capturing the realism of traffic flow propagation, there have been shortcomings in the existing node models. Tampère et al. (2011) have studied the necessary requirements that a comprehensive node model should meet and have accordingly proposed a generic first order node model which meets all of the essential criteria to generate realistic and oriented capacity-proportional distribution of the available downstream supply over the incoming links of a node. This proposed node model is transferable to multi-commodity flow and can be used for any node configuration including simple merges or diverges, general nodes with multiple merges and diverges, etc. These characteristics make the Tampère et al. (2011) node model suitable for applications in traffic assignment and network loading models. This node model has been implemented in the QDNL algorithm by Bliemer et al. (2014) and has, as well, been used in the system optimal traffic assignment model presented in this paper.

The system optimal traffic assignment problem seeks to find an optimal traffic flow pattern for the network, such that the total network cost (travel time) is minimized. Potential applications include traffic management, congestion pricing, evacuation planning, and work zone and incident traffic management. A large body of research in the past decade that deals with the system optimal traffic assignment problems concentrates on single-destination system optimal dynamic traffic assignment (SD-SODTA). The SD-SODTA problem seeks to optimize the traffic routing from multiple sources (origins) to a single sink (destination), with dynamic flows and travel times. "Single-destination" optimal traffic routing formulations have been accepted and widely applied to evacuation optimization problems in the past decade (Chiu et al., 2007; Nassir, 2013; Shen, 2009; Zheng, 2009). Modeling evacuation in a single-destination network facilitates the use of many efficient solution algorithms that have been developed for network flow problems, such as Minimum Cost Dynamic Flow (Nassir, Zheng, et al., 2014), Earliest Arrival Flow (Zheng et al., 2013), Quickest Flow (Fleischer & Skutella, 2007; Fleischer, 2001) for SD-SODTA problems such as Exit Flow Function models (Carey, 1987; Merchant & Nemhauser, 1978a; Merchant & Nemhauser, 1978b; Nie, 2011), or SD-SODTA formulations modeled with the Cell Transmission Model (CTM) (Ukkusuri et al., 2009; Ziliaskopoulos, 2000). The underlying assumption of a single-destination optimal evacuation routing is that all vehicles in the network are routed from the nodes inside the threat area to safe locations outside of the threat area. With this assumption, a general network could be transformed into a single-destination network by virtually connecting all of the safe locations to a dummy super-sink and assigning that super-sink as the single destination of the network. The single-destination characteristic allows for efficient graph theoretic and linear programming algorithms for computationally efficient solutions, however, the disadvantage is that it only applied to single-destination networks.

From the perspective of formulating a system optimal traffic assignment problem, there exist two main approaches, namely link-based and path-based. The link-based formulations seek to optimize turn flows from upstream links to alternative downstream links, whereas the path-based formulations seek to optimize the distribution of origin-destination (O-D) demands among predefined path alternatives. Since the path sets sizes grow combinatorial with respect to number of network links, a link-based formulation leads to smaller number of decision variables and is thus more tractable. There are computationally efficient Linear Programming (LP) formulations that have been implemented to model and solve the link-based SO problem. However, in the originally proposed link-based formulations, the constraint sets

were mainly non-linear (Merchant & Nemhauser, 1978a; Merchant & Nemhauser, 1978b) and the modifications to make the constraints linear (Carey, 1987; Ziliaskopoulos, 2000) may lead to issues such as flow holding on links, especially in many-to-many networks. The flow holding issue in a system optimal dynamic traffic assignment model can happen when, in a solution, despite positive remaining capacity in the downstream, vehicles are being held at an upstream link or a junction, to accommodate other path flows that are possibly more critical to the system objective. Although the objective value may be superior in a flow holding solution, such a solution is considered impractical and undesirable, because it is usually assumed that vehicles do not selflessly hold or slow down to favor the system objective. In order to eliminate such instances additional constraints should be incorporated. In addition, more detailed path flow constraints (such as the first-in-first-out rule – FIFO) are difficult to represent in link-based formulations. On the other hand, path-based formulations can easily capture these constraints, but at the expense of higher computational burden.

A path-based SO traffic assignment problem can be converted into a Variational Inequality and be solved using conventional traffic assignment solution finding algorithms. The key to this solution is the estimation of PMC. PMC is defined as the changes in total system travel time due a single unit of flow perturbation on each individual path. In the SO static assignment problem, at the optimum solution, the PMC on all used paths connecting a given O-D pair are equal to or less than the PMC on any unused paths (Peeta & Mahmassani, 1995). Similarly for SODTA, Peeta and Mahmassani (1995) proved that, at the optimum solution, the time-dependent PMC on all used paths connecting a given O-D pair are equal to or less than the time-dependent PMC on any unused paths. As a result, assigning vehicles to paths with the minimum PMC between each O-D pair will lead to a SODTA solution.

In typical *static* traffic assignment problems where travel times are obtained through explicit volume-delay functions (e.g. the BPR function), marginal costs are simply calculated by taking the derivative of the link travel time functions with respect to flow. However, in more advanced *dynamic* and *quasi-dynamic* traffic assignment problems that use network loading procedures, path travel times are computed implicitly and thus the estimation of PMC is not a trivial task. Therefore, designing efficient algorithms to approximate the PMC -without enumerating and loading one perturbed solution for every path- can be very valuable in solving SO traffic assignment problems. Examples of such studies in the context of path-based SODTA are as follows.

Peeta and Mahmassani (1995) utilized PMC to obtain the SODTA solution for general networks. They obtained PMC by summing up link marginal costs (obtained as the derivative of the time-dependent link performance function) along the paths, with the consideration of link traversal times. This method was later shown to overestimate the PMC (Shen et al., 2007). Ghali and Smith (1995) also estimated an SODTA solution using link marginal costs in general networks. Shen et al. (2007) estimated PMC by tracing the changes in link cumulative flow arrival and departure curves along the paths, for the special case of networks without diverges. Later, Qian et al. (2012) used a similar approach and generalized the model to include diverges as well.

In the context of Quasi-Dynamic Traffic Assignment (QDTA) however, little has been done to design efficient PMC estimation algorithms to be used in System Optimal Quasi-Dynamic Traffic Assignment (SOQDTA). Despite the advantages of the aforementioned studies, there still exists the need for efficient system optimal traffic assignment models that can be solved for many-to-many real-size networks, fast enough to suit real-time applications. The above review of the existing literature highlights the possibility to take advantage of, and build upon the existing state-of-the-art QDNL models to develop a computationally efficient system

optimal traffic assignment framework which can assist many traffic management applications.

Given the computational and practical benefits of the newly-proposed QDNL algorithm by Bliemer et al. (2014), we propose a PMC approximation algorithm consistent with this QDNL algorithm, to be used for SOQDTA. The resultant path-based SOQDTA problem can be solved for many-to-many networks without issues such as flow-holding.

Methodology

In this section, we further discuss the SOQDTA problem, its objective function and constraints, as well as the embedded components, namely the node model, the QDNL module, the PMC estimation algorithm and the assignment procedure. We have adopted the QDNL model by Bliemer et al. (2014), which incorporates the node model by Tampère et al. (2011). Accordingly, we have proposed a PMC approximation algorithm to align with these founding models.

The following path-based SOQDTA problem seeks to find the optimal path flow values f_p that lead to minimized total system travel time, subject to demand and traffic flow constraints:

$$\min \mathrm{TC}(\mathbf{f}) = \sum_{(r,s)} \sum_{p \in P^{rs}} f_p * C_p(\mathbf{f}) \quad (1)$$

$$\text{s.t.} \quad \sum_{p \in P^{rs}} f_p = D^{rs} \ , \forall (r,s)$$

$$f_p \geq 0 \ , \forall p \in P^{rs}, \forall (r,s)$$

where $C_p(\mathbf{f})$ represents the average travel time on path p, with $\mathbf{f}$ being the vector of network path flows pattern consisting of path flows for all O–D pair paths, D^{rs} represents the total demand between the O-D pair (r,s), where r belongs to R, the set of all origin nodes, and s belongs to S the set of all destination nodes. In the above equations, P^{rs} represents the set of all paths between each O-D pair (r,s).

According to Wardrop's second principle (Wardrop, 1952), in path-based SO traffic assignment solutions, the PMC on all the used paths among each O-D pair are equal, and less than or equal to the PMC of any unused paths between the same O-D. Thus the SO assignment solution can generally be obtained by assigning vehicles to paths with the least marginal cost (least PMC) between each O-D. The problem of finding the paths with the least PMC has been referred to as the least marginal cost problem in the literature (Peeta & Mahmassani, 1995; Qian et al., 2012; Shen et al., 2007).

By definition, $PMC_p(\mathbf{f})$ is equal to the derivate of total system travel time with respect to flow on path p and under flow pattern $\mathbf{f}$. In dynamic traffic assignment models, $PMC_p(\mathbf{f})$ is conventionally computed by adding one unit of flow on path p (unit perturbation on path p) and measuring the total change in the system travel time. However, the perturbation size does not necessarily have to be unit, and in this paper we have tested different values of perturbation size in a sensitivity analysis which is conducted in the case study experiment.

The total system travel time value $TC(\mathbf{f})$ can be re-written as the summation of total path travel times (travel time experienced by all vehicles on any path) over all the paths of the network, regardless of the O-D pairs, as:

$$TC(\mathbf{f}) = \sum_{i=1}^{|P|} TC_{p_i} = \sum_{i=1}^{|P|} f_{p_i} * C_{p_i}(\mathbf{f}) \quad (2)$$

where $P = \cup_{(r,s)} P^{rs}$ indicates the set of all network paths.

Having equation (2) the gradient vector of $TC(\mathbf{f})$ or the vector of ***PMC*(f)** can be computed as:

$$\boldsymbol{PMC}(\mathbf{f}) = \frac{\partial TC(\mathbf{f})}{\partial \mathbf{f}} = \begin{bmatrix} \frac{\partial TC_{p_1}}{\partial f_{p_1}} & \frac{\partial TC_{p_2}}{\partial f_{p_1}} & \cdots & \frac{\partial TC_{p_{|P|}}}{\partial f_{p_1}} \\ \frac{\partial TC_{p_1}}{\partial f_{p_2}} & \frac{\partial TC_{p_2}}{\partial f_{p_2}} & \cdots & \frac{\partial TC_{p_{|P|}}}{\partial f_{p_2}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial TC_{p_1}}{\partial f_{p_{|P|}}} & \frac{\partial TC_{p_2}}{\partial f_{p_{|P|}}} & \cdots & \frac{\partial TC_{p_{|P|}}}{\partial f_{p_{|P|}}} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} PMC_{p_1}(\mathbf{f}) \\ PMC_{p_2}(\mathbf{f}) \\ \vdots \\ PMC_{p_{|P|}}(\mathbf{f}) \end{bmatrix} \quad (3)$$

Any perturbation of flow on each path p_i will affect the total system travel time in two ways, one being the effect on the travel time of vehicles on the same path p_i, and the other being the effect on the vehicles on the other paths that interact with path p_i. The impact of perturbation of path p_i on the same path is referred to as *internality* and is represented by $I_{p_i}(\mathbf{f}) = \frac{\partial TC_{p_i}}{\partial f_{p_i}}$.The total impact of the perturbation of path p_i on all other paths, is referred to as *externality* and is represented by $E_{p_i}(\mathbf{f}) = \sum_{p_j \in P, j \neq i} \frac{\partial TC_{p_j}}{\partial f_{p_i}}$. The summation of each paths' internality and externality return the path's marginal cost, as in equation (3).

The $PMC_{p_i}(\mathbf{f})$ of any path p_i, can be written as follows:

$$PMC_{p_i}(\mathbf{f}) = \frac{\partial TC_{p_i}}{\partial f_{p_i}} + \sum_{p_j \in P, j \neq i} \frac{\partial TC_{p_j}}{\partial f_{p_i}} = \underbrace{\left(f_{p_i} * \frac{\partial C_{p_i}(\mathbf{f})}{\partial f_{p_i}} + C_{p_i}(\mathbf{f}) \right)}_{I_{p_i}(\mathbf{f})} + \underbrace{\left(\sum_{p_j \in P, j \neq i} \frac{\partial C_{p_j}(\mathbf{f})}{\partial f_{p_i}} * f_{p_j} \right)}_{E_{p_i}(\mathbf{f})} \quad (4)$$

Computation of $PMC_{p_i}(\mathbf{f})$ using the equation (4) requires the computation of path travel times ($C_{p_i}(\mathbf{f})$) and partial derivatives of path travel times with respect to path flows ($\frac{\partial C_{p_j}(\mathbf{f})}{\partial f_i}$). These values need to be defined according to the underlying assumptions of the QDNL algorithm and are introduced at length in the following subsections.

The framework in Figure 1 demonstrates the overall procedure for generating a SOQDTA solution and specifies how the different elements of QDNL, the node model and the PMC estimation are incorporated in this procedure. Initially, in the network and demand specification step, the link lengths and maximum speeds, the O-D demands and time horizon *T* are input to the problem. The network is considered to be a directed graph *G(N,A)*, where *N* and *A* denote the set of nodes and links respectively. Link capacities are also denoted by $\Theta_a, \forall a \in A$. Next, a set of feasible paths is generated for every non-zero demand O-D pair using a path set generation algorithm. The Path-Size Penalty Algorithm (PSPA) is used here (Nassir, Ziebarth, et al., 2014) to generate the set of reasonable path alternatives, but not necessarily in an increasing order of travel times (or distances). The PSPA generates the set of path alternatives that are reasonable (in terms of travel time) and sufficiently independent. We generate a maximum of 10 alternative paths for every OD pair, as a fixed a-priori set. Ideally, path set generation can be repeated after the network loading step at every iteration of the algorithm so that new relevant paths are discovered at every iteration. However, we have used an a-priori set of reasonable and independent paths to simplify the computations.

The SOQDTA module uses the method of successive averages (MSA) to assign the O-D demands to paths such that minimal total system travel time is achieved. The MSA, simple but effective, has been widely used in traffic assignment problems (Sheffi & Powell, 1982). It starts with a feasible solution in the solution space, (all-or-nothing assignment of the O-D demand to an initially shortest path alternative) and then revises this solution by shifting proportions of the demand to the paths with the current least cost (least PMC here). At each iteration of the MSA, the least marginal cost problem is solved for every O-D pair and a new assignment flow pattern is generated and loaded onto the network, and repeated until convergence is reached. For completeness, the MSA algorithm used in this study is explained in Appendix A.

We will first present an overview of the two adopted founding models and then elaborate on our proposed PMC approximation algorithm.

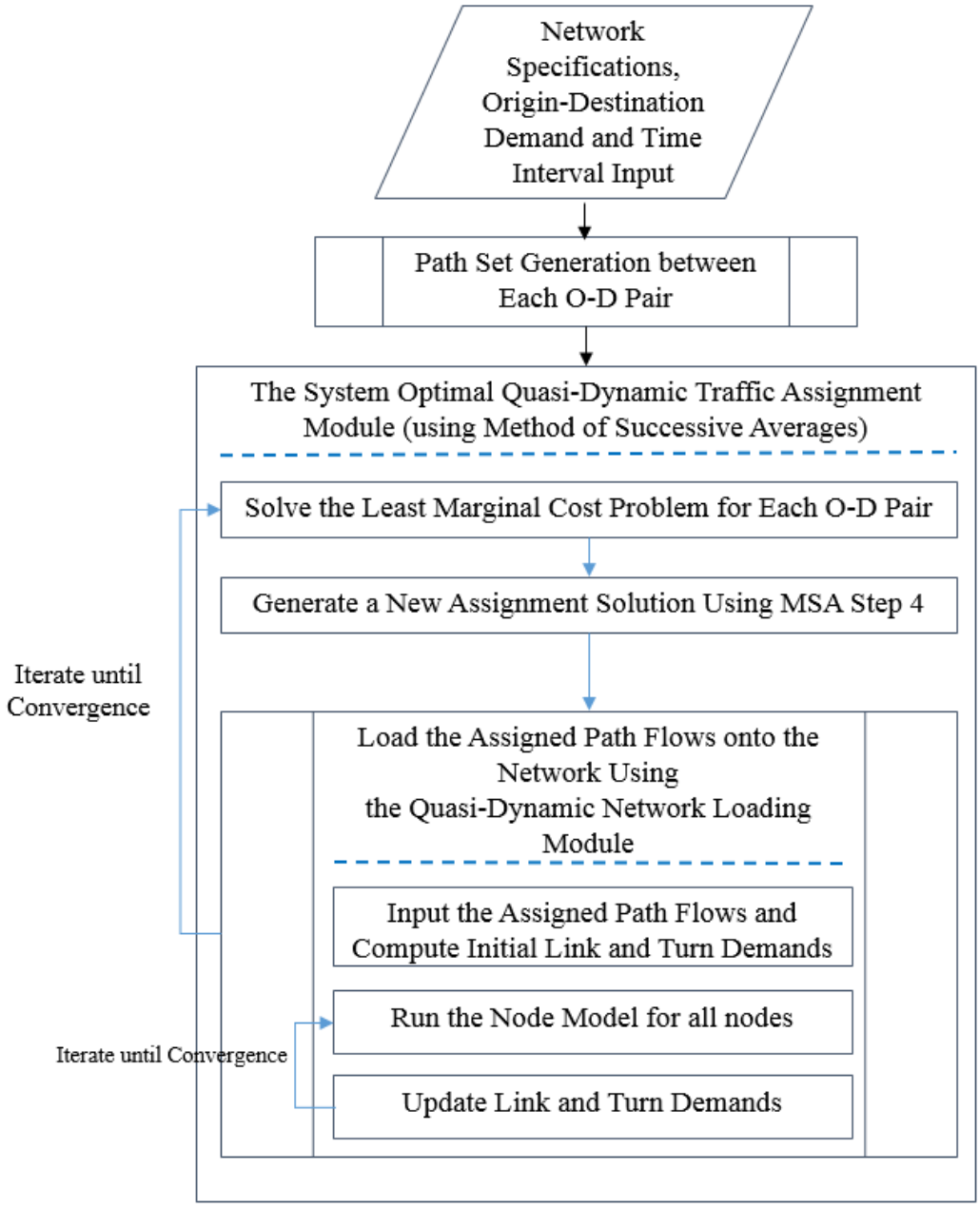

Figure 1- The Overall SOQDTA Methodological Framework

- The QDNL model and The First-Order Node Model

The QDNL procedure takes the path flows f_p (generated by the assignment module) as input and computes the link flows and turn flows as output. This QDNL model incorporates a first order node model that computes link reduction factors α_a^{basic} for all links, as the ratio of the link actual out-flow to the link in-flow (link demand).

All links with reduction factors smaller than one are considered congested because of a downstream bottleneck. Reduction factors are then applied to the path flows to compute link and turn flows. In order to guarantee the FIFO requirement, only a unique value of α_a^{basic} is associated with each link *a* to all its out-going turn directions. In other words, for a given link *a* incoming to node *n* or $a \in A_n^{in}$, $\alpha_{a-b}^{basic} = \alpha_a^{basic}, \forall b \in A_n^{out}$, where A_n^{out} represents the set of out-links of node *n* and A_n^{in} represents the set of in-links to node *n*. This condition guarantees that regardless of the downstream link congestion conditions, the out-flow rate from one link is equal for all the downstream links.

The link reduction factors are mathematically defined as:

$$\alpha_a^{basic} = \frac{q_a}{S_a} = \frac{q_{a-b}}{S_{a-b}} \qquad \forall\, a \in A_n^{in}, \forall b \in \{A_n^{out} | S_{a-b} > 0\}, \forall n \in N \tag{5}$$

$$q_a = \sum_{b \in A_n^{out}} q_{a-b} \qquad \forall\, a \in A_n^{in}, \forall n \in N \tag{6}$$

where S_a denotes the total in-flow to link *a* or demand on link *a*, $q_a$denotes the total actual out-flow from link *a*, S_{a-b} denotes the turn demand from link *a* to *b*, and q_{a-b} denotes the actual turn flow from link *a* to *b*.

It should be noted that potential queue spillback has not been considered in this model, as it leads to non-stationary flows and contradicts the assumption of steady-state flows.

Following Bliemer et al. (2014) and Bifulco and Crisalli (1998) path-specific link in-flows can be computed using link reduction factors.

$$S_{ap} = \delta_{ap} f_p \prod_{a' \in \eta_{ap}} \alpha_{a'}^{basic} \qquad \forall\, a \in A, \forall\, p \in P^{rs}, \forall (r,s) \tag{7}$$

where η_{ap} denotes the set of links on path *p* from the origin up to, but not including, link *a*, and δ_{ap} is the link-path incidence indicator.

The turn demands can also be computed as follows:

$$S_{a-b} = \sum_{(r,s)} \sum_{p \in P^{rs}} \delta_{bp} S_{ap} \qquad \forall\, a \in A_n^{in}, \forall b \in A_n^{out}, \forall n \in N \tag{8}$$

The total link in-flow of link *a* will then be computed as the sum of path-specific link in-flows (S_{ap}) over all paths using link *a*. The value of S_a is also equal to the sum of turn demands from link *a* to all downstream turn directions.

$$S_a = \sum_{(r,s)} \sum_{p \in P^{rs}} S_{ap} = \sum_{b \in A_n^{out}} S_{a-b} \qquad \forall a \in A_n^{in}, \forall n \in N \tag{9}$$

As evident from equations (7) to (9), the link and turn flows are determined using link reduction factors α_a^{basic} obtained from the node model. Thus, in a more general representation, $\boldsymbol{S'} = \Upsilon(\boldsymbol{\alpha^{basic}}|\mathbf{f})$; where $\boldsymbol{S'}$ denotes the vector of all turn demands ($\boldsymbol{S'} = [S_{a-b}]$) and depends on $\boldsymbol{\alpha}$ and $\mathbf{f}$ which respectively denote the vector of all reduction factors ($\boldsymbol{\alpha^{basic}} = [\alpha_a^{basic}]$) and the vector of all path demand flows ($\boldsymbol{f} = [f_p]$).

On the other hand, the node model requires the link and turn flows and capacities of all incoming and outgoing links of a node as input in order to compute $\boldsymbol{\alpha}^{\boldsymbol{basic}}$. Generally $\boldsymbol{\alpha}^{\boldsymbol{basic}}$ can be obtained as a function of link turn demands given link capacities as follows:

$$[\alpha_a^{basic}]_{a\in A_n^{in}} = \boldsymbol{\Gamma}^{\boldsymbol{n}}(S_{a'-b'}|\Theta_{a'},\Theta_{b'}) \qquad \forall a' \in A_n^{in}, \forall b' \in A_n^{out} \tag{10}$$

where $\boldsymbol{\Gamma}^{\boldsymbol{n}}(.)$ is an implicit representation of the node model.

This circular reference between $\boldsymbol{\alpha}$ and $\boldsymbol{S}$ yields the following fixed point problem:

$$\boldsymbol{\alpha}^{\boldsymbol{basic}} = \boldsymbol{\Gamma}(\boldsymbol{S}'|\Theta) = \boldsymbol{\Gamma}\big(\Upsilon(\boldsymbol{\alpha}^{\boldsymbol{basic}}|\mathbf{f})\,|\Theta\big) = \boldsymbol{g}(\boldsymbol{\alpha}^{\boldsymbol{basic}}|\boldsymbol{f},\Theta) \tag{11}$$

The solution to this fixed point problem is a vector of reduction factors $\boldsymbol{\alpha}^{\boldsymbol{basic}^*}$ which satisfies $\boldsymbol{\alpha}^{\boldsymbol{basic}^*} = \boldsymbol{g}(\boldsymbol{\alpha}^{\boldsymbol{basic}^*}|\boldsymbol{f},\Theta)$, where $\boldsymbol{g} = \boldsymbol{\Gamma} \circ \boldsymbol{\Upsilon}$.

Bliemer et al. (2014) propose the following algorithm to iteratively solve this fixed point problem:

Input: Path set $\boldsymbol{P}$, path flows $\mathbf{f}^{(i)}$, where i represents the assignment iteration, and link capacities $\boldsymbol{\theta}$.

Step 0: *Initialization.* Assuming an empty network, initialize all reduction factors $\boldsymbol{\alpha}^{\boldsymbol{basic}^{(0)}} = 1$.

Step 1: *Calculate initial link and turn flows.* For all paths $p \in \boldsymbol{P}$, calculate path-specific link in-flows using $\mathbf{f}^{(i)}$ (equation 7). Calculate turn demands $\boldsymbol{S}'^{(\mathbf{0})}$ (equation 8), and calculate link in-flows $\boldsymbol{S}^{(\mathbf{0})}$ (equations 9). Set the loading iteration $j = 1$.

Step 2: *Determine potentially congested links*[1]. For each link $b \in A$, if $S_b^{(0)} > \Theta_b$ then link b is a potential bottleneck and all links a with turn demand $S_{a-b}^{(0)} > 0$ will potentially be congested. In other words, the set of links $\tilde{A}$ that will be considered for QDNL is defined as: $\tilde{A} = \{a \in A_n^{in} |\ S_{a-b}^{(0)} > 0, S_b^{(0)} > \Theta_b, b \in A_n^{out}, n \in N\}$

Step 3: *Compute Reduction Factors.* Run the node model using turn demands $\widetilde{\boldsymbol{S}}'^{(j-1)}$, link flows $\tilde{\boldsymbol{S}}^{(j-1)}$ from previous iteration (*j-1*), to obtain reduction factors $\widetilde{\boldsymbol{\alpha}}^{\boldsymbol{basic}^{(j)}}$.

Step 4: *Compute turn and link demands*. Compute the path-specific link flows using $\widetilde{\boldsymbol{\alpha}}^{\boldsymbol{basic}^{(j)}}$ and $\mathbf{f}^{(i)}$ (equation 7) and calculate the turn demands $\widetilde{\boldsymbol{S}}'^{(j)}$ (equation 8). Use turn demands (equation 9) to update link flows $\tilde{\boldsymbol{S}}^{(j)}$ for all $a \in \tilde{A}$.

Step 5: *Convergence check*. If $\frac{1}{|\tilde{A}|}\left\|\widetilde{\boldsymbol{\alpha}}^{\boldsymbol{basic}^{(j)}} - \widetilde{\boldsymbol{\alpha}}^{\boldsymbol{basic}^{(j-1)}}\right\| < \varepsilon_1$, for convergence criteria parameter $\varepsilon_1 > 0$, the problem has converged to a fixed point, then terminate. Otherwise, set $j = j + 1$ and return to step 3. ($\|\cdot\|$ Represents Euclidean Norm)

The resultant reduction factors from the QDNL are then used to compute the average path travel times as follows:

$$C_p(\mathbf{f}) = \sum_{a\in A} \frac{\delta_{ap} L_a}{v_a^{max}} + \frac{T}{2}\left(\frac{1}{\prod_{a\in V^p} \alpha_a^{basic}} - 1\right) \tag{12}$$

[1] This step is performed in order to enhance the computational efficiency of the algorithm. By this means, the links that are potentially congested will be identified and for the links that will potentially not be congested α_a^{basic} will remain equal to one.

where $\sum_{a\in A}\frac{\delta_{ap}L_a}{v_a^{max}}$ denotes the summation of links free flow travel times, [0,T] denotes the demand time period duration, V^p denotes the vector of links on path p and

$\frac{T}{2}\left(\frac{1}{\prod_{a\in V^p}\alpha_a^{basic}}-1\right)$ denotes the average non-separable path queuing delay, in consistence with queueing theory (Bliemer et al., 2014).

The first order node model presented here is adapted from Tampère et al. (2011) and can be applied to general cross-nodes. Their proposed algorithm finds the exact solution in a maximum of m iterations (m being the number of node in-links). This node model can be extended for signalized intersections, using the signal green time ratios per turn. We refer to Tampère et al. (2011) for the detailed node-model algorithm.

- Analytical PMC Derivation

In order to solve the path-based SOQDTA problem defined in equation (1), PMCs need to be approximated for all paths between all O-D pairs. In order to compute path internality and externality using equation (4), the values of $\frac{\partial C_{p_i}(\mathbf{f})}{\partial f_{p_i}}$ and $\frac{\partial C_{p_j}(\mathbf{f})}{\partial f_{p_i}}$ $(where\ j\neq i)$ need to be approximated respectively. By plugging the path travel times from equation (9) into equation (4), $\frac{\partial C_{p_i}(\mathbf{f})}{\partial f_{p_i}}$ can be written as follows:

$$\frac{\partial C_{p_i}(\mathbf{f})}{\partial f_{p_i}} = \frac{T}{2}*\frac{-1}{(\prod_{a\in V^{p_i}}\alpha_a^{basic})^2}*\left(\sum_{b\in V^{p_i}}\frac{\partial\alpha_b^{basic}}{\partial f_{p_i}}*\frac{\prod_{a\in V^{p_i}}\alpha_a^{basic}}{\alpha_b^{basic}}\right)$$
$$= \frac{T}{2}*\frac{-1}{\prod_{a\in V^{p_i}}\alpha_a^{basic}}*\left(\sum_{b\in V^{p_i}}\frac{\partial\alpha_b^{basic}}{\partial f_{p_i}}*\frac{1}{\alpha_b^{basic}}\right) \quad (13)$$

Since according to equation (11), $\boldsymbol{\alpha^{basic}}$ are calculated through the node model and the fixed point QDNL problem and not through an explicit function, deriving the analytical partial derivate of alphas with respect to path flows is not trivial. As a result, we propose to approximate $\frac{\partial\alpha_b^{basic}}{\partial f_{p_i}}$ by enumerating $\frac{\Delta\alpha_b^{basic}}{\Delta f_{p_i}}=\frac{\alpha_{b,p_i}^{new}-\alpha_b^{basic}}{\xi_{V_1^{p_i},p_i}}$; then,

$$\frac{\partial C_{p_i}(\mathbf{f})}{\partial f_{p_i}} = \frac{T}{2}*\frac{-1}{\prod_{a\in V^{p_i}}\alpha_a^{basic}}*\left(\sum_{b\in V^{p_i}}\frac{\alpha_{b,p_i}^{new}-\alpha_b^{basic}}{\xi_{V_1^{p_i},p_i}}*\frac{1}{\alpha_b^{basic}}\right) \quad (14)$$

where α_{b,p_i}^{new} denotes the new reduction factor of link b as a result of perturbation on path p_i, and $V_k^{p_i}$ denotes the k^{th} link on the path p_i. Also, $\xi_{V_1^{p_i},p_i}$ denotes the initial perturbation on the first link of path p_i when path p_i is perturbed.

Similarly, $\frac{\partial C_{p_j}(\mathbf{f})}{\partial f_{p_i}}, \forall j\neq i$ can be written as follows:

$$\frac{\partial C_{p_j}(\mathbf{f})}{\partial f_{p_i}} = \frac{T}{2}*\frac{-1}{\prod_{a\in V^{p_j}}\alpha_a^{basic}}*\left(\sum_{b\in V^{p_j}}\frac{\alpha_{b,p_i}^{new}-\alpha_b^{basic}}{\xi_{V_1^{p_i},p_i}}*\frac{1}{\alpha_b^{basic}}\right) \quad (15)$$

In the proposed method for computation of externality, two forms of approximation take place. First, we only approximate externality on paths that share at least one node with path p_i, and for simplicity, we assume that perturbation on path p_i will not affect those paths of the network that do not share any nodes with path p_i. Second, in estimation of externality on

an intersecting path, we only consider the link reduction factor changes for the immediate incoming link to the shared node.

- PMC Approximation Algorithm

According to the overall SOQDTA framework in Figure 1, a feasible path flow pattern (solution) will be generated by the assignment module and will be loaded onto the network using the QDNL module. Afterwards, the PMC values should be updated according to the newly obtained flow pattern on the network.

From the QDNL converged solution, link demands $\boldsymbol{S}$, turn demands (flows) $\boldsymbol{S'}$ and link reduction factors $\boldsymbol{\alpha^{basic}}$ are available. In the QDNL model, path travel times are calculated based on the reduction factors of path links (equation (9)) and thus the effects of one additional unit of demand on travel time can be captured through the changes that it causes in link reduction factors. In turn, reduction factors are determined through the node model which captures the complicated interactions between link and turn demands and capacities.

In the first step of PMC approximation, regardless of paths, at every node $n \in N$, we evaluate the potential changes in the in-link reduction factors $\alpha_a^{basic}, a \in A_n^{in}$ due to one extra unit of flow on different turn movements $m_{a,b}, \forall\, a \in A_n^{in}, \forall b \in A_n^{out}, \forall n \in N$. More specifically, one perturbation unit is added to each turn demand S_{a-b} (obtained from the converged QDNL solution), at each node *n* in the network as follows:

$$S_{a-b}^{+1} = S_{a-b} + 1 \qquad a \in A_n^{in}, b \in A_n^{out}, n \in N \tag{16}$$

For this stage, only one turn movement $m_{a,b}$ is perturbed at a time, keeping the other turn demands constant. Next, the node model is run for node *n* with the new turn demands (S_{a-b}^{+1}), and the corresponding reduction factors on in-links a', denoted by $\alpha_{a',m_{a,b}}^{+1}$ $\forall a' \in A_n^{in}$, are calculated. This stage is performed for all nodes and their existing turn movements in the network, regardless of the paths using these movements. Therefore, the computational cost of this stage increases only linearly with the number of nodes in the network. Moreover, since at this stage, each node is processed individually, parallel processing can also be performed for higher efficiency.

The relative changes between $\alpha_{a',m_{a,b}}^{+1}$ and $\alpha_{a'}^{basic}$ are then stored in variable $\Delta\alpha_{a',m_{a,b}}^{+1}$, $\forall a' \in A_n^{in}$, as follows:

$$\Delta\alpha^{+1}{}_{a',m_{a,b}} = \frac{\alpha_{a',m_{a,b}}^{+1} - \alpha_{a'}^{basic}}{\alpha_{a'}^{basic}}, \qquad \forall\, \alpha' \in A_n^{in}, \forall b' \in A_n^{out} \tag{17}$$

Figure 2 gives a graphical demonstration of this stage in a node with two in-links and two out-links. The procedure can be applied to any general cross-node and we have merely limited the number of in and out links in the figure for illustration clarity.

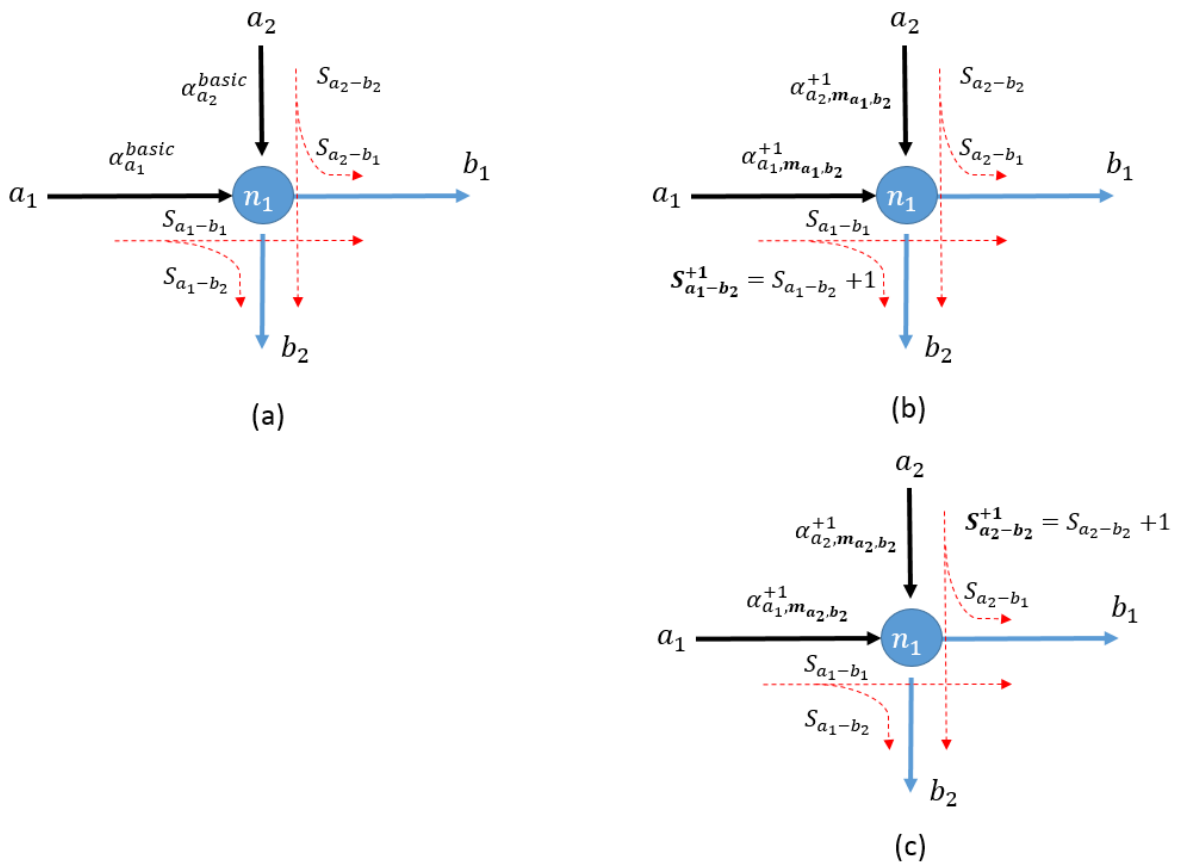

Figure 2- Computation of $\boldsymbol{\alpha}^{+1}$ for a general cross-node

Figure 2 (a) demonstrates a general cross-node n_1, where the turn demands and link reduction factors have been computed in the QDNL procedure. In Figure 2 (b), we have added one demand unit to movement m_{a_1,b_2} from link a_1 to b_2. The turn demand $S_{a_1-b_2}$ will be increased by one unit, and other turn demands remain constant. This demand increase on one movement affects all incoming link reduction factors. This effect is captured using the node model and by calculating new alphas, $\alpha_{a_1,\boldsymbol{m_{a_1,b_2}}}^{+1}$ and $\alpha_{a_2,\boldsymbol{m_{a_1,b_2}}}^{+1}$. Figure 2 (c) shows one additional unit of demand on movement m_{a_2,b_2}, and same should be performed for m_{a_1,b_1} and m_{a_2,b_1}.

So far, the aforementioned steps compute and store the relative changes to $\boldsymbol{\alpha^{basic}}$ due to one unit of increased demand on each movement. The next step in the algorithm takes the generated O-D paths into account and perturbs each path individually to approximate its internal and external effects on total system travel time. This stage of the algorithm also allows for parallel processing of different O-D pairs.

In this stage, each path p is perturbed by a fixed additional flow $\xi_{V_1^p,p}$ that is assigned on path p at its first link V_1^p, and the estimated proportion of this unit flow, $\xi_{a,p}$ that reaches every downstream link a is computed using link reduction factor. $\xi_{a,p}$ is then used to approximate the resultant $\boldsymbol{\alpha^{new}}$ on all the links of the network that are affected by perturbing path p. In the original version of the algorithm we choose the perturbation size at the first link $\xi_{V_1^p,p}$=1, however, we have also tested different values for $\xi_{V_1^p,p}$ in a sensitivity analysis that is reported in the cases study section.

In a general cross-node, when a given link a is an incoming link to node n ($a \in A_n^{in}$), according to the node model, the perturbation arriving on link a from path p ($\xi_{a,p}$) affects the reduction factors of **all** incoming links to node n ($\forall\, a' \in A_n^{in}$). Since changes in reduction factors with respect to unit perturbation ($\Delta\boldsymbol{\alpha}^{+1}$) are computed in the previous step of the algorithm, a linear approximation can provide the correspondent $\boldsymbol{\alpha}^{new}$ with respect to $\boldsymbol{\xi} = [\xi_{a,p}]$.

$$\alpha_{a',p}^{new} = \left(1 + \xi_{a,p} * \Delta\alpha_{a',m_{a,b}}^{+1}\right) * \alpha_{a'}^{basic} \qquad \forall a' \in A_n^{in}, a \in A_n^{in}, b \in A_n^{out}, n \in N^p \qquad (18)$$

In equation (18) $\alpha_{a',p}^{new}$ denotes the new reduction factor of link a' as a result of perturbation on path p, and N^p denotes the set of all nodes on path p. However, due to potential bottlenecks, the unit flow may not proceed all the way to destination (during unit time interval). The proportion of unit perturbation that will reach any link a on the path is computed as:

$$\xi_{a,p} = \xi_{V_1^p,p} * \prod_{l \in \eta_{ap}} \alpha_{l,p}^{new} \qquad \forall a \in V^p, \forall p \in P^{rs}, \forall (r,s) \qquad (19)$$

where $\alpha_{l,p}^{new}$ denotes the new reduction factors of links l, resultant from perturbing path p. Equation (19) is based on equation (7) in which the product of reduction factors upstream of a link determines the link in-flow.

The algorithm processes all movement $m_{a,b}$ on the path sequentially starting from the upstream ($\forall\, m_{a,b} \in M^p$), where M^p is the set of all movement on path p and a and b are consecutive links on path p ($a = V_i^p, b = V_{i+1}^p, a \in A_n^{in}, b \in A_n^{out}, n \in N^p$), and computes $\boldsymbol{\alpha}^{new}$, for not only the links on the path ($a \in V^p, a \in A_n^{in}$) but also all other links a' that share a node with links a ($\forall\, a' \in A_n^{in}$).

Figure 3 illustrates this stage of the algorithm:

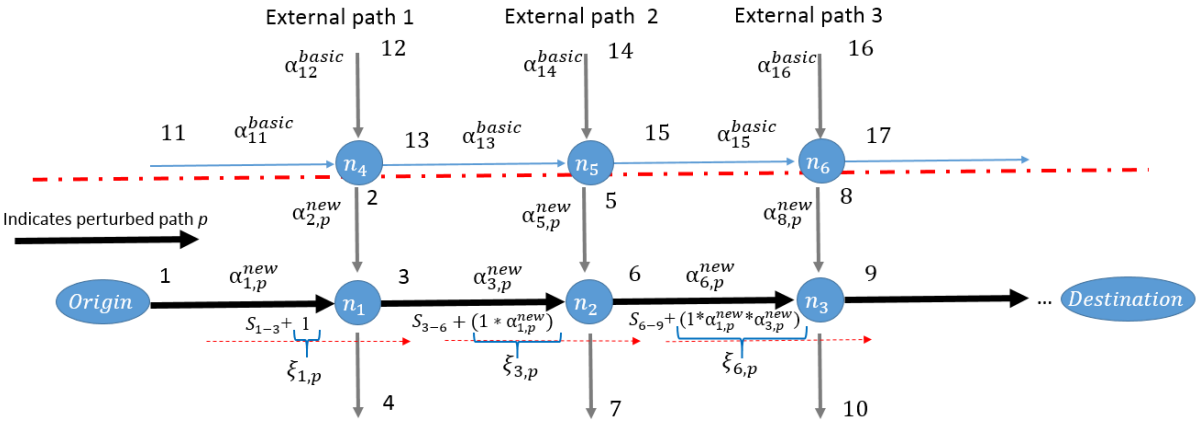

Figure 3- Computation of $\boldsymbol{\alpha}^{new}$ for links affected by perturbation on path p

As the algorithm processes each movement $m_{a,b}$ on the path p, path externality at each node $n \in N^p$ is computed for all paths p' that share the same node with path p.

The internality of path p and the total externality can be computed once all the movements on the path are processed. Examples of external paths to path p are demonstrated in Figure 3, as external paths 1, 2, and 3. While processing each node n on path p ($n \in N^p$), the impact of the perturbation $\xi_{a,p}$ can be evaluated on all the external links incoming to node n as follows:

$$\lambda_{a',p'}^{n,p} = \frac{T}{2} * \frac{-1}{\prod_{a \in V^{p'}} \alpha_a^{basic}} * \left(\frac{\alpha_{a',p}^{new} - \alpha_{a'}^{basic}}{\xi_{V_1^p,p}} * \frac{1}{\alpha_{a'}^{basic}} \right) * f_{p'}$$

$$\forall a' \in A_n^{in}, \forall p' \in P_{a'}, p' \neq p, n \in N^p \quad (20)$$

Where $\lambda_{a',p'}^{n,p}$ denotes the share of externality happening on link a' of path p' when perturbation reaches node *n* on path *p*. This value shall be computed for all O-D paths using link a', a set denoted by $P_{a'}$.

In Figure 3, external path 1, as an example, shares node n_1 with path *p*. The algorithm approximates the impact of path *p* on this external path by computing $\alpha_{2,p}^{new}$ and $\lambda_{2,1}^{n_1,p}$. However the reduction factor on link 12, which is on the external path but is not an immediate in-link to node n_1, remains constant at α_{12}^{basic}. The reason behind this lies in the assumptions of the QDNL model, where queues are assumed to form vertically (point queues) and potential spillback is not explicitly modelled.

The red dashed line in Figure 3, shows the extent to which the impact of path *p* on the external paths 1, 2 and 3 is taken into account.

Once all the movements on path *p* are processed, the total externality of the path can be computed as:

$$E_p(\mathbf{f}) = \sum_{n \in N^p} \sum_{a' \in A_n^{in}} \sum_{p' \in P_{a'} \setminus p} \frac{T}{2} * \frac{-1}{\prod_{a \in V^{p'}} \alpha_a^{basic}} * \left(\sum_{a' \in A_n^{in}} \frac{\alpha_{a',p}^{new} - \alpha_{a'}^{basic}}{\xi_{V_1^p,p}} * \frac{1}{\alpha_{a'}^{basic}} \right) * f_{p'} \quad (21)$$

Equation (21) accounts for the approximations that take place in the computation of path externality, namely that externality is computed only for paths that share at least a node with path *p*, and the assumption that only certain one link per external path is affected by the perturbation on path *p*.

The internality of path *p* can also be computed using equations (4) and (14), once all the movements have been processed and the reduction factors of all links on path *p* have been updated. Having computed path internality and externality, the total PMC can be obtained.

The pseudocode for the procedure explained above, can be found in Appendix B.

Computational Advantages

At every iteration of the search for the least marginal cost paths, the proposed PMC approximation initially computes and stores the changes in the link reduction factors due to unit perturbations on all existing movements at all nodes ($\Delta\boldsymbol{\alpha}^{+\mathbf{1}}$), one movement at a time. This computation is done only once at every iteration and independent of the O-D paths. Therefore, the computational cost of this stage increases only linearly with the size of network. Moreover, since nodes are processed individually, parallel processing becomes possible for higher efficiency.

Next, for each O-D pair, the algorithm computes PMC_p ($\mathbf{f}$) by moving along every O-D path p, looking up the stored $\Delta\boldsymbol{\alpha}^{+\mathbf{1}}$ and accordingly approximating the travel time of path p and all crossing paths. Without this store-and-look-up approximation technique, PMC approximation would entail moving along each O-D path and running the node model for every node along the path with perturbed link demands. As a matter of fact, we are reducing the computational time significantly by factoring out the procedure of solving the node models for every path, and performing this process only once and independently of the number of paths.

To compute the externality of one path, p, at every iteration of the MSA, the proposed algorithm needs to estimate changes in travel times of all (and only) crossing paths p'. This will require computation of $\dot{A}_{p',p}^{new}$ for all crossing paths p'.

Considering the above-mentioned steps for the PMC calculations, the required number of computations are bound to O(|N| + |P|·|A| + $|P|^2$|A|) which results from |N| (number of node in the network) times executions of the node model at the beginning, |P|·|A| (number of paths × number of links) times update of link reduction factor for internality of paths, and $|P|^2$|A| path travel time update for externalities of every path on every intersecting path. Because this upper bound is polynomial in the size of the network, the execution of the method for real-sized networks is possible. However, for fast computations in large networks, parallel computations may be required.

Case Study

In order to evaluate the performance of the proposed SOQDTA algorithm, we have applied it to a test network of the medium-sized city of Sioux Falls. This network has been used in transportation studies several times (Lam & Zhang, 2000; LeBlanc et al., 1975; Morlok, 1973), to evaluate the performance of different traffic assignment models. The specific demand table of the Sioux Falls network, used in this study, was originally used by LeBlanc et al. (1975) and is accessible via the website Transportation Test Problems (Bar-Gera).

The Sioux Falls test network has 24 nodes and 76 links. The O-D demand matrix used in this study represents the all-to-all node demand for one hour during the peak period with a total of 360,600 trips. A total of 3,125 paths have been generated for 528 non-zero O-D demand pairs, using the path set generation algorithm by Nassir, Ziebarth, et al. (2014).

In order to provide evidence that the SO model is actually improving the objective function value compared to the UE solution, we have computed the user equilibrium quasi-dynamic traffic assignment (UEQDTA) and SOQDTA solutions for the network and compared the total system travel times (objective values) at each MSA iteration. The changes in link reduction factors, as a measure of links out-flow to in-flow ratios, have also been demonstrated.

We have used the MSA (presented in Appendix A) with one hundred iterations, to compute both the UEQDTA and SOQDTA solutions. In step 2 of the MSA algorithm, the O-D paths with the least PMC will be chosen to add flow in the SOQDTA problem; whereas in the same step, the O-D paths with the least travel times are chosen to add flow for the UEQDTA problem.

The values of total system travel time under UE and SO assumptions have been demonstrated over the MSA iterations in Figure 4.

Like all quasi-dynamic models with residual queues, our model also assumes no queues at the beginning of the time interval (Bliemer et al., 2014), and is thus applied best to the whole peak period. Accordingly, all PMCs are equal to the path free flow travel times at the first assignment iteration, and the total system travel time is initially equal for UEQDTA and SOQDTA. However, in the next MSA iterations the SOQDTA constantly leads to lower total system travel times compared to UEQDTA. After one hundred iterations, total system travel times are $1.26 * 10^7$and $1.2 * 10^7$ minutes under UE and SO, respectively, demonstrating a total improvement of about 4.8%.

Coded in C++, on a notebook computer with Intel® Core™ i7 @ 2.6 GHz running Windows 7 32-bit, using a single thread, the computational time of each MSA iteration is 0.48 second for UEQDTA, and 3.50 seconds for the proposed SOQDTA. The 3.50 seconds for SOQDTA include PMC approximation, solving the least marginal cost problem, finding the new assignment solution and loading the solution onto the network (the QDNL).

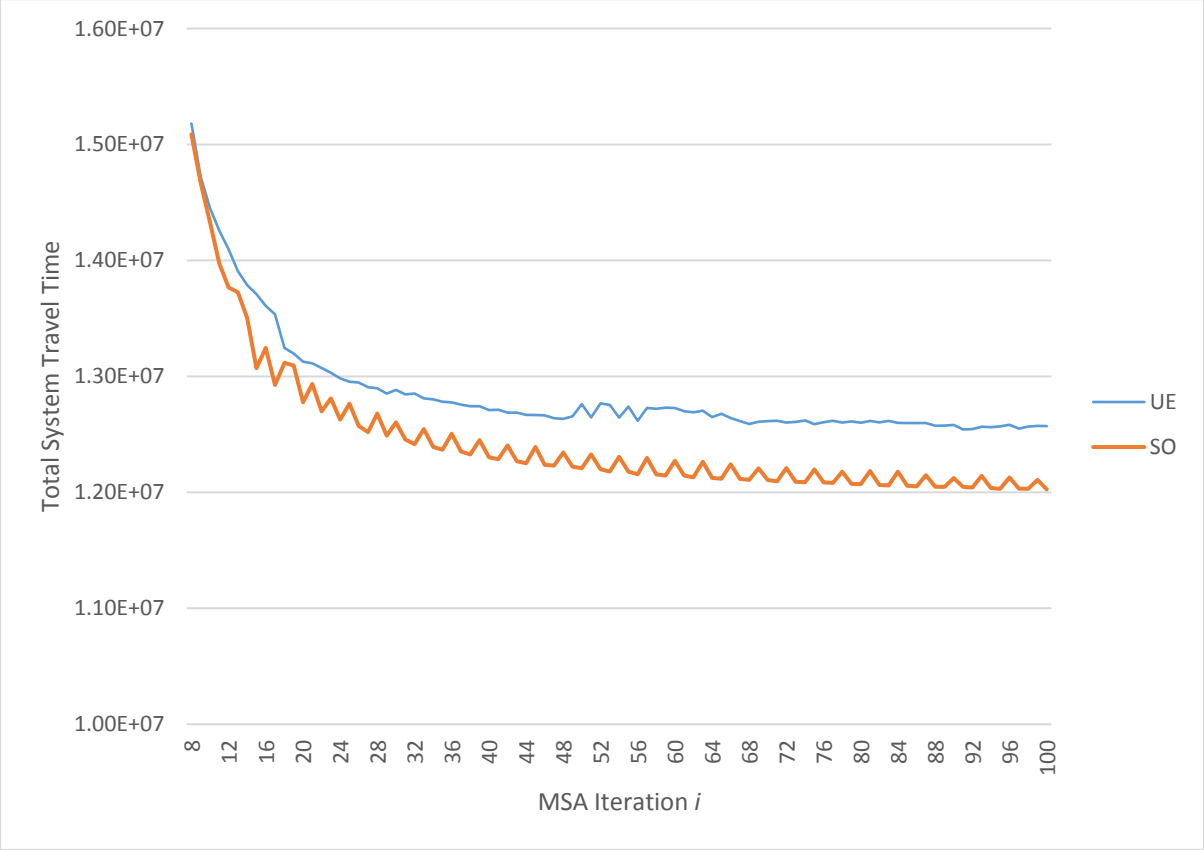

Figure 4-SOQDTA vs. UEQDTA Total System Travel Time (minutes) for the Test Network of Sioux Falls

The fluctuation of the SO objective value along the MSA iterations observed in Figure 4 is because: 1) the descent direction computed based on PMCs is approximate, and 2) the MSA step size in the descent direction is not necessarily optimized. However, the SO trend line is

demonstrating an overall decreasing pattern and SO objective values are constantly lower than the UE. This observation demonstrates that moving in the direction of the approximated PMCs leads to an improved total system travel time (objective value); hence the algorithm is capable of providing a reliable and practical approximation of PMCs. Future research may focus on modifying the step size in descent direction in order to improve the efficiency of SO computation.

As previously discussed, according to Wardrop's second principle (Wardrop, 1952), in path-based SO traffic assignment solutions, the PMC on all the used paths among each O-D pair are equal, and less than or equal to the PMC of any unused paths between the same O-D. The SO relative gap function value is thus defined using the following equation for each assignment iteration i:

$$G^{(i)} = \frac{\sum_{\forall(r,s)} \sum_{\forall p \in P^{rs}} f_p * (PMC_p(\mathbf{f}) - \mu^{rs})}{\sum_{\forall(r,s)} \sum_{\forall p \in P^{rs}} f_p * \mu^{rs}} \quad (22)$$

where $\mu^{rs} = min_{p \in P^{rs}} PMC_p(\mathbf{f})$, $\forall(r,s)$.

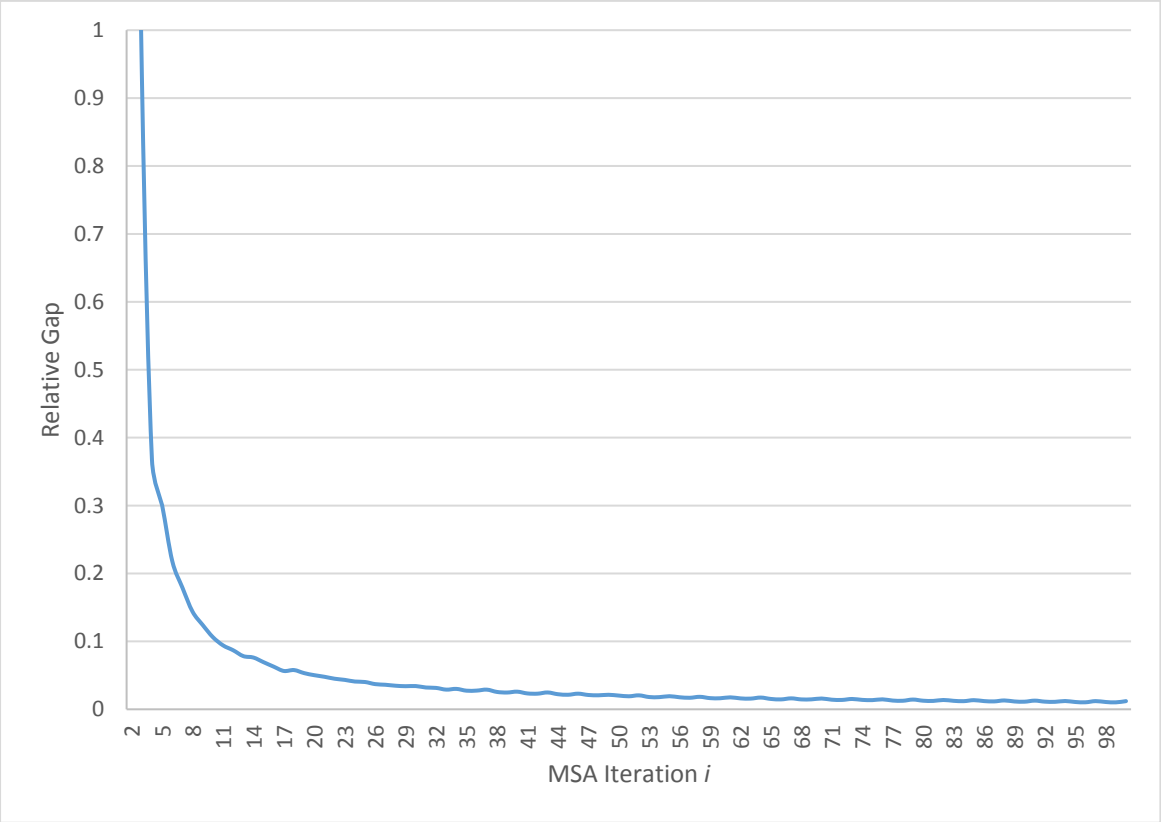

Figure 5- Relative Gap Function Value at each MSA Iteration

As depicted in Figure 5, the relative gap reaches 0.01 after 100 MSA iterations, demonstrating that the algorithm is converging to a SO solution.

Figure 6 demonstrates the changes made in link reduction factors between the UEQDTA and SOQDTA solutions. The SOQDTA solution increases link reduction factors on some links and decreases them on some other, for a generally improved total system travel time. To elaborate on specific examples in Figure 6, the SO link reduction factors increased on links 9, 22, 39, 40, 43, 52, 59, 61, and 68, and decreased on links 12 and 63.

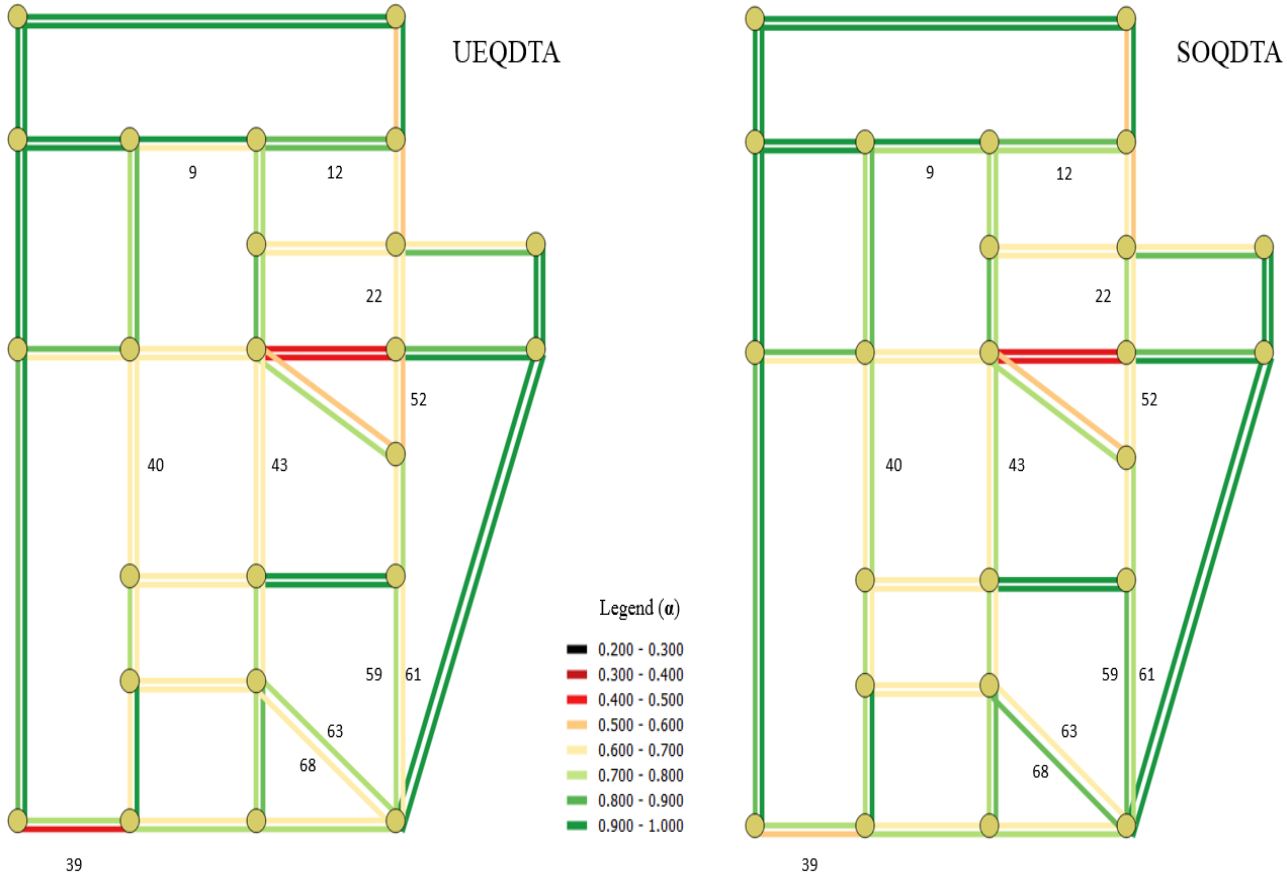

Figure 6- Link Reduction Factors under UE and SOQDTA

A sensitivity analysis has been performed on the perturbation size, for values ranging between 0 and 5, in intervals of 0.1 units (Figure 7). It is observed that the optimal objective value achieved by different values of perturbation is generally increasing (worsening) in a step-like function. This is consistent with intuition, as increasing the perturbation size, should make the PMC approximation less accurate due to the fact that smaller (than perturbation size) changes in O-D flows, which could potentially yield a better objective value, may be undetected in the PMC approximation with a not small enough perturbation size.

In this particular case study, we observe that as long as the perturbation size is smaller than 1.6, the algorithm will perform efficiently in 100 MSA iterations. In general, given that the flow changes in MSA are proportional to the O-D demand and inversely proportional to the iteration number, a sufficiently small perturbation size should be computed based on O-D demands and the maximum number of iterations that MSA is run for.

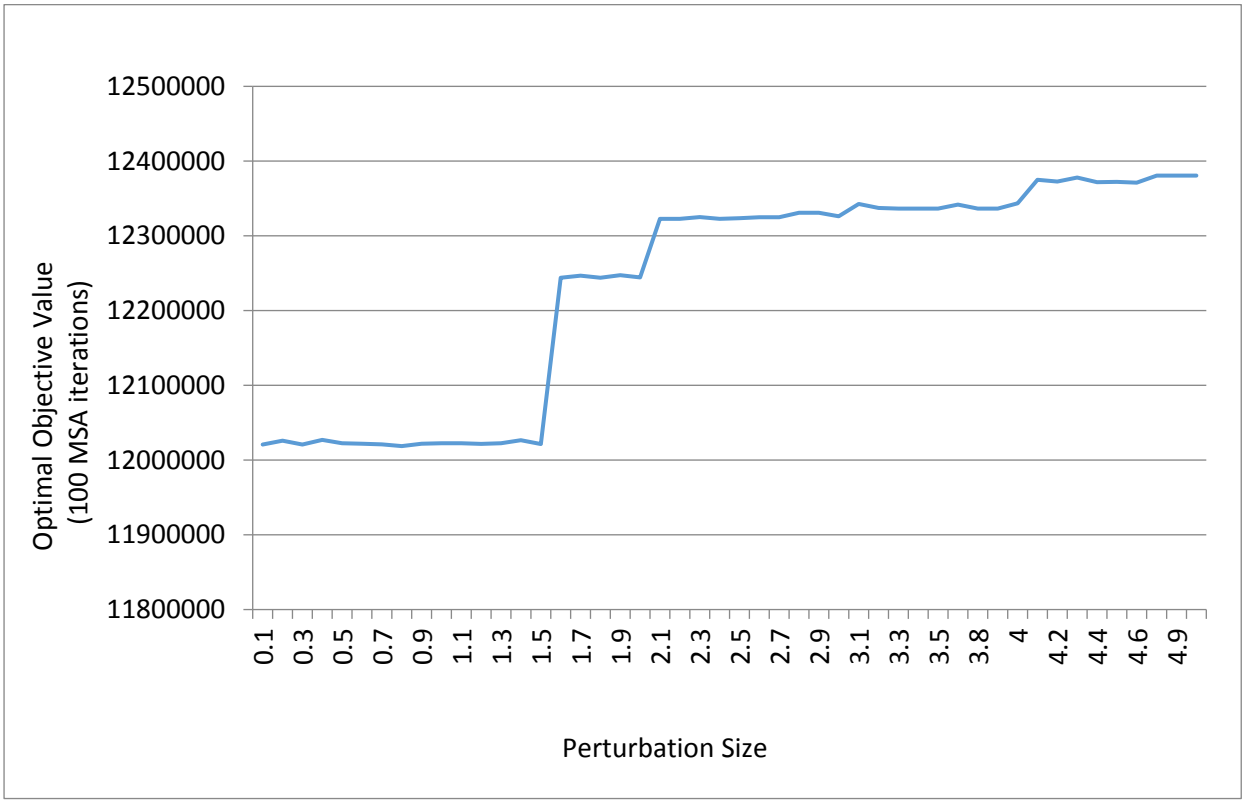

Figure 7- Optimal Objective Value Obtained after 100 MSA Iterations using different Values of Perturbation

Conclusions and Future Research

We have proposed, developed and tested a computationally efficient path-based SOQDTA framework, and a new PMC approximation algorithm for solving it. The framework incorporates a QDNL model that captures the realism of traffic propagation through a first-order node model.

The proposed quasi-dynamic model benefits the computational efficiency of static models, yet considers capacity constraints, residual vertical/point queues and observes the FIFO principle. Being path-based, the model does not lead to issues such as flow holding in the network, and can be efficiently solved for many-to-many networks.

The resultant SO traffic flow pattern can have a variety of applications from regular traffic management to work zones and incident traffic re-routing and disaster evacuation. The corresponding minimum total system travel time also provides a benchmark for evaluating potential benefits of different traffic management strategies.

We have applied our proposed model to Sioux-falls test network. The results demonstrate that the model is capable of providing reliable estimates of the PMC and can be used for SO traffic management. The obtained SO solution demonstrated an improvement of 4.8% in total system travel time compared to the UE solution. The values of the relative gap function were also reported which demonstrated the convergence at 0.01 level after 100 MSA iterations.

A sensitivity analysis of the perturbation size was also performed and demonstrated that the optimal objective value achieved by different values of perturbation is generally increasing (worsening) in a step-like function. Therefore, the perturbation size should be chosen small enough to guarantee the solution quality. In general, given that the flow changes in MSA are proportional to the O-D demand and inversely proportional to the iteration number, the perturbation size should be selected based on O-D demand levels and the maximum number of iterations.

As a possible direction for future research, we identify testing and analysis of different solution finding algorithms, which in the current model is done by a generic MSA. Adjustments of the step size in the solution finding process is of particular interest and expected to further improve both the computational efficiency and the quality of solution of the proposed SOQDTA.

To further complement the present research from the perspective of implementation, one can explore practical implications of applying the proposed algorithms to work zones traffic management or incident traffic re-routing scenarios. Especially, the recent advancements in connected and autonomous vehicle technologies, and new possibilities for advisory traffic information provision, can open up new opportunities to reconcile the state-of-the-art in SO traffic modelling with the state-of-practice in traffic management.

The present study does not account for traffic signals. The incorporation of signals through capacity adjustments will be also considered in future research.

References

Bar-Gera, H. Transportation Test Problems. Retrieved from https://github.com/bstabler/TransportationNetworks

Bifulco, G., & Crisalli, U. (1998). Stochastic user equilibrium and link capacity constraints: formulation and theoretical evidence. In *TRANSPORTATION PLANNING METHODS. PROCEEDINGS OF SEMINAR E HELD AT AET EUROPEAN TRANSPORT CONFERENCE, LOUGHBOROUGH UNIVERSITY, UK, 14-18 SEPTEMBER 1998. VOLUME P424*.

Bliemer, M. C., Raadsen, M. P., Smits, E.-S., Zhou, B., & Bell, M. G. (2014). Quasi-dynamic traffic assignment with residual point queues incorporating a first order node model. *Transportation Research Part B: Methodological, 68*, 363-384.

Carey, M. (1987). Optimal Time-Varying Flows on Congested Networks. *Operations Research, 35*(1), 58-69.

Chiu, Y.-C., Zheng, H., Villalobos, J., & Gautam, B. (2007). Modeling No-notice Mass Evacuation Using a Dynamic Traffic Flow Optimization Model. *IIE Transactions, 39*(1), 83-94.

Fleischer, L., & Skutella, M. (2007). Quickest Flows Over Time. *SIAM Journal on Computing, 36*(6), 1600-1630.

Fleischer, L. K. (2001). Faster Algorithms for the Quickest Transshipment Problem. *SIAM Journal on Optimization, 12*(1), 18-35.

Ghali, M., & Smith, M. (1995). A model for the dynamic system optimum traffic assignment problem. *Transportation Research Part B: Methodological, 29*(3), 155-170.

Lam, W. H., & Zhang, Y. (2000). Capacity-constrained traffic assignment in networks with residual queues. *Journal of transportation engineering, 126*(2), 121-128.

LeBlanc, L. J., Morlok, E. K., & Pierskalla, W. P. (1975). An efficient approach to solving the road network equilibrium traffic assignment problem. *Transportation Research, 9*(5), 309-318.

Merchant, D. K., & Nemhauser, G. L. (1978a). A model and an algorithm for the dynamic traffic assignment problems. *Transportation science, 12*(3), 183-199.

Merchant, D. K., & Nemhauser, G. L. (1978b). Optimality Conditions for a Dynamic Traffic Assignment Model. *Transportation Science, 12*(3), 200-207.

Morlok, E. K. (1973). *Development and application of a highway network design model*: Federal Highway Administration.

Nassir, N. (2013). *Optimal Integrated Dynamic Traffic Assignment and Signal Control for Evacuation of Large Traffic Networks with Varying Threat Levels* PhD Dissertation. University of Arizona, Tucson.

Nassir, N., Zheng, H., & Hickman, M. (2014). Efficient negative cycle-canceling algorithm for finding the optimal traffic routing for network evacuation with nonuniform threats. *Transportation Research Record: Journal of the Transportation Research Board*(2459), 81-90.

Nassir, N., Ziebarth, J., Sall, E., & Zorn, L. (2014). Choice Set Generation Algorithm Suitable for Measuring Route Choice Accessibility. *Transportation Research Record: Journal of the Transportation Research Board*(2430), 170-181.

Nie, Y. M. (2011). A Cell-Based Merchant-Nemhauser Model for the System Optimum Dynamic Traffic Assignment Problem. *Transportation Research Part B: Methodological, 45*(2), 329-342.

Peeta, S., & Mahmassani, H. S. (1995). System optimal and user equilibrium time-dependent traffic assignment in congested networks. *Annals of Operations Research, 60*(1), 81-113.

Qian, Z. S., Shen, W., & Zhang, H. (2012). System-optimal dynamic traffic assignment with and without queue spillback: Its path-based formulation and solution via approximate path marginal cost. *Transportation research part B: methodological, 46*(7), 874-893.

Sheffi, Y., & Powell, W. B. (1982). An algorithm for the equilibrium assignment problem with random link times. *Networks, 12*(2), 191-207.

Shen, W. (2009). *System Optimal Dynamic Traffic Assignment: A Graph-Theoretic Approach and its Engineering Application* (Ph.D. thesis). University of California, Davis, Davis, CA.

Shen, W., Nie, Y., & Zhang, H. M. (2007). On path marginal cost analysis and its relation to dynamic system-optimaltraffic assignment. *TRANSPORTATION AND TRAFFIC THEORY 2007: PAPERS SELECTED FOR PRESENTATIONAT ISTTT17, A PEER REVIEWED SERIES SINCE 1959*(p327-60).

Tampère, C. M., Corthout, R., Cattrysse, D., & Immers, L. H. (2011). A generic class of first order node models for dynamic macroscopic simulation of traffic flows. *Transportation Research Part B: Methodological, 45*(1), 289-309.

Ukkusuri, S. V., Ramadurai, G., & Patil, G. (2009). A Robust Transportation Signal Control Problem Accounting for Traffic Dynamics. *Computers & Operations Research, 37*(5), 869-879.

Wardrop, J. G. (1952). ROAD PAPER. SOME THEORETICAL ASPECTS OF ROAD TRAFFIC RESEARCH. *Proceedings of the institution of civil engineers, 1*(3), 325-362.

Zheng, H. (2009). *Efficient Algorithms for the Cell-Based Single Destination System Optimal Dynamic Traffic Assignment Problem* Ph.D.(Ph.D. thesis). University of Arizona, Tucson, AZ.

Zheng, H., Chiu, Y.-C., & Mirchandani, P. B. (2013). A Heuristic Algorithm for the Earliest Arrival Flow with Multiple Sources. *Journal of Mathematical Modelling and Algorithms in Operations Research*, 1-21.

Ziliaskopoulos, A. K. (2000). A linear programming model for the single destination system optimum dynamic traffic assignment problem. *Transportation science, 34*(1), 37-49.

Appendix A

MSA algorithm for SOQDTA (Qian et al., 2012):

Step 0: Initialization. Select an initial flow pattern $\boldsymbol{f}^{\beta}$,$\beta = 0$.

Step 1: Perform the QDNL based on $\boldsymbol{f}^{\beta}$.

Step 2: Solve the least marginal cost problem for each O-D pair *(r,s)*. $p_{rs}^{*} = argmin_{p} PMC_{p}^{rs}(\mathbf{f})$

Step 3: Obtain the auxiliary flow pattern $\boldsymbol{g}(\boldsymbol{f}^{\beta})$ by assigning all the demand $D^{rs}\forall(r,s)$ onto p_{rs}^{*}.

Step 4: $\boldsymbol{f}^{\beta+1} = \boldsymbol{f}^{\beta}\left(1-\frac{1}{\beta}\right) + \boldsymbol{g}(\boldsymbol{f}^{\beta})\frac{1}{\beta}$

Step 5: Convergence check, if $\left\|\boldsymbol{f}^{\beta+1} - \boldsymbol{f}^{\beta}\right\| < \varepsilon$ terminate. Otherwise go to step 1 and set $\beta = \beta + 1$.

Appendix B

Algorithm to approximate PMC

- From the QDNL solution, obtain the turn demands $\boldsymbol{S'}$, and the link reduction factors $\boldsymbol{\alpha^{basic}}$
- For all nodes *n* in the network:
  - For all turn movements $m_{a,b}$ at node *n*:
    - Add 1 perturbation unit to turn demand on $m_{a,b}$ (keeping the other turn demands unchanged)
    - Run the node model and calculate all $\alpha_{a',m_{a,b}}^{+1}$, $\forall a' \in A_n^{in}$
    - Store the relative changes to $\alpha_{a'}^{basic}$(as a result of unit perturbation on movement $m_{a,b}$) in variable $\Delta\alpha_{a',m_{a,b}}^{+1}$, $\forall a' \in A_n^{in}$
- For all O-D pairs (r, s):
  - For all paths $p \in P^{rs}$:
    - Initiate perturbation of size 1 on the first link of the path, $\xi_{V_1^p,p} = 1$;
    - For all movements on path *p*, $m_{a,b} \in M^p$, $(a = V_i^p, b = V_{i+1}^p, a \in A_n^{in}, b \in A_n^{out},\ n \epsilon N^p)$:
      - For all in-links to node *n*, $a' \in A_n^{in}$:
        - Compute $\alpha_{a',p}^{new} = \left(1 + \xi_{a,p} * \Delta\alpha_{a',m_{a,b}}^{+1}\right) * \alpha_{a'}^{basic}$
          $(a' \in A_n^{in}, a = V_i^p, b = V_{i+1}^p)$
      - For all paths p' that use links a', $\forall a' \in A_n^{in}, \forall p' \in P_{a'}$ && $p' \neq p$:
        - Compute the share of externality happening on link a' of path p' when perturbation reaches node *n* on path *p*
          $\lambda_{a',p'}^{n,p} = \frac{T}{2} * \frac{-1}{\prod_{a \in V^{p'}} \alpha_a^{basic}} * \left(\frac{\alpha_{a',p}^{new} - \alpha_{a'}^{basic}}{\xi_{V_1^p,p}} * \frac{1}{\alpha_{a'}^{basic}}\right) * f_{p'}$
          $(\forall a' \in A_n^{in}, \forall p' \in P_{a'}, p' \neq p, n \in N^p)$
      - Update the value of perturbation that can pass through link *a*, on path *p*, and reach the next link on the path, according to α_a^{new}
        $\xi_{b,p} = \xi_{a,p} * \alpha_{a,p}^{new}$ $(a = V_i^p, b = V_{i+1}^p)$
    - Compute Total Externality of path *p:*

$$E_p(\mathbf{f}) = \sum_{n \in N^p} \sum_{a' \in A_n^{in}} \sum_{p' \in P_{a'} \setminus p} \frac{T}{2} * \frac{-1}{\prod_{a \in V^{p'}} \alpha_a^{basic}} * \left(\sum_{a' \in A_n^{in}} \frac{\alpha_{a',p}^{new} - \alpha_{a'}^{basic}}{\xi_{V_1^p,p}} * \frac{1}{\alpha_{a'}^{basic}} \right) * f_{p'}$$

- Compute Internality on path *p*:

$$I_p(\mathbf{f}) = f_p * \left(\frac{T}{2} * \frac{-1}{\prod_{a \in V^p} \alpha_a^{basic}} * \left(\sum_{b \in V^p} \frac{\alpha_{b,p}^{new} - \alpha_b^{basic}}{\xi_{V_1^p,p}} * \frac{1}{\alpha_b^{basic}} \right) \right) + C_p(\mathbf{f})$$

- Compute marginal cost of path *p*:

$$PMC_p(\mathbf{f}) = I_p(\mathbf{f}) + E_p(\mathbf{f})$$