Netw Spat Econ (2010) 10:49–71
DOI 10.1007/s11067-007-9022-y

Solving the Dynamic User Optimal Assignment Problem Considering Queue Spillback

Yu (Marco) Nie · H. M. Zhang

Published online: 1 June 2007

Abstract This paper studies the dynamic user optimal (DUO) traffic assignment problem considering simultaneous route and departure time choice. The DUO problem is formulated as a discrete variational inequality (DVI), with an embeded LWR-consistent mesoscopic dynamic network loading (DNL) model to encapsulate traffic dynamics. The presented DNL model is capable of capturing realistic traffic phenomena such as queue spillback. Various VI solution algorithms, particularly those based on feasible directions and a line search, are applied to solve the formulated DUO problem. Two examples are constructed to check equilibrium solutions obtained from numerical algorithms, to compare the performance of the algorithms, and to study the impacts of traffic interacts across multiple links on equilibrium solutions.

Keywords Dynamic user optimal traffic assignment · Dynamic network loading · Variational inequality · Feasible direction algorithms

1 Introduction

Predicting the temporal and spatial traffic evolution over road networks has attracted numerous research efforts since the 1970s, when transportation researchers began to recognize the limitations of static network equilibrium models (Wardrop 1952, Beckmann et al. 1956) in describing a system that essentially varies over time. These limitations include, among others, the inability of modeling departure time decisions and the formation and dissipation of queues when travel demands temporarily exceed road capacities at different locations, two important elements to consider in traffic congestion management.

Y.(M.) Nie (✉)
Department of Civil and Environmental Engineering, Northwestern University, Evanston, IL, USA
e-mail: y-nie@northwestern.edu

H. M. Zhang
Department of Civil and Environmental Engineering, University of California,
Davis, CA 95616, USA
e-mail: hmzhang@ucdavis.edu

The earliest work to bring the time dimension into equilibrium analysis is perhaps Vickrey's seminal paper on the morning commute problem (Vickrey 1969), in which traffic congestion takes the form of queuing behind a bottleneck caused by temporal demand surge. Vickrey's bottleneck model stimulated numerous studies on the morning commute problem. Most of these studies focused on finding analytical equilibrium solutions and/or exploring economic and policy insights (e.g., Mahmassani and Herman 1984, Newell 1987, Arnott et al. 1990, Kuwahara 1990, Yang and Huang 1997). However, these models do not intend to solve, thereby are not typically applicable to the general dynamic user optimal (DUO) assignment problem, where multiple origin–destination (O–D) pairs, route/departure time choices and realistic representation of traffic dynamics are considered simultaneously.

It was not until the early 1990s that variational inequality (VI) is employed to provide a general DUO formulation, in the context of dynamic traffic assignment (DTA). Friesz et al. (1993) and Smith (1993) are the first to establish the equivalence between the dynamic user optimal condition and the solution to a corresponding variational inequality VI($\mathbf{c}$, Ω), where $\mathbf{c}$ represents a mapping from path flow to path cost and Ω depicts a feasible set of path flows. Nevertheless, the original VI formulation of Friesz et al. (1993) is notoriously hard to solve. Above all, the analytical properties of the nested path cost function $\mathbf{c}$, such as differentiability and monotonicity, are difficult to assert (Friesz et al. 2001). This drawback hinders the design of effective solution procedures. Numerous formulations had since been proposed in the hope of casting the flow-cost mapping in a more tractable way. Most existing work are either based on VI (e.g., Wei et al. 1995, Ran et al. 1996, Chen and Hsueh 1998, Lo and Szeto 2002, Szeto and Lo 2004), or highly related to it (e.g., Huang and Lam 2002, Wei et al. 2002). In this body of work, various network traffic flow models were used, which include the delay function model (e.g., Friesz et al. 1993, Ran et al. 1996), the point-queue model (e.g., Smith 1993, Huang and Lam 2002), and the kinematic wave model (e.g., Lo and Szeto 2002, Szeto and Lo 2004). In spite of these efforts, however, the evaluation of $\mathbf{c}$ still relies frequently on a numerical procedure known as dynamic network loading (DNL). The complexity of DNL varies substantially according to the underlying representation of traffic flow, which conceivably has a dominant impact on the DUE solutions. Compared to DNL, designing solution procedures for dynamic traffic assignment is less well studied: heuristic algorithms were suggested in most cases, such as the method of successive averages (Tong and Wong 2000). Lo and Szeto (2002) solved the DUO problem using an alternating direction algorithm. By introducing Lagrangian multipliers, they simplified the projection used to generate the feasible direction. Recently, Friesz and Mookherjee (2006) proposed a projection algorithm which converges to a DUO solution when the nested path cost function is monotonic.[1]

In this paper, the DUO problem is formulated as a discrete VI while traffic dynamics is encapsulated in a mesoscopic DNL model. The formulation considers individuals' route and departure time choices simultaneously. Our DNL model describes traffic evolution on links through macroscopic traffic models based on the kinematic wave theory, and considers link interactions by incorporating proper node models. It is thus capable of capturing realistic traffic phenomena such as the propagation of shockwaves and queue spillback.

[1] It is noted (Friesz and Mookherjee 2006), however, that this condition is unlikely to be met in general.

We examine a class of numerical solution procedures for the DUO problem and use various line search techniques to improve convergence performance. Most of these techniques are based on transforming a VI to a maximization program associated with a bi-variate merit function. The use of line search seems prohibitively expensive at first glance since evaluating the merit function would at least require performing DNL once. Existing feasible direction algorithms often apply predetermined step sizes (e.g., Szeto and Lo 2004). However, our experiments indicate that the computational overhead invested in finding a better step size could be paid off by the gains of faster convergence.

It is known that a traffic flow model ignoring the physical space taken up by queues may undermine the prediction accuracy of DTA models. This paper further shows that just considering the physical length of queues is still not enough: equilibrium solutions may vary considerably if the propagation speed of queues (shockwave speed) is computed differently, as is made clear in Section 5. This finding again attests to the importance of a realistic representation of traffic dynamics in modeling network flows.

2 A VI formulation of the DUO problem

Consider a general transportation network $G(N, A)$, where N and A are sets of nodes and links, respectively. Let R and S represent the set of origins and destinations, K^{rs} the set of paths joining an O–D pair rs. Let $[0, T]$ be an *assignment period*. The network is assumed to be empty at $t=0$, and only vehicles departing within the assignment period are of interest to the DUO problem. Corresponding to each assignment period, we define a *loading period* $[0, T']$, where T' marks the time when all traffic clears the network. Let $q^{rs}(t)$ be the travel demand between O–D pair rs departing at time t, and

$$q^{rs} = \int_0^T q^{rs}(t)$$

Without loss of generality, we assume that all commuters prefer a punctual arrival within an identical time window, $\left[t_a^* - \delta, t_a^* + \delta\right]$. A schedule cost is imposed if they fail to do so. Each individual would choose a combination of a departure time t and a route p^{rs} such that the *actual experienced* commute cost, $c_p^{rs}(t)$, is minimized. We define $c_p^{rs}(t)$ by the following piecewise linear function (Arnott et al. 1990):

$$c_p^{rs}(t) = \begin{cases} \alpha\omega_p^{rs}(t) + \beta\left[t_a^* - \Delta - t - \omega_p^{rs}(t)\right], & t + \omega_p^{rs}(t) < t_a^* - \Delta \quad \text{early arrival} \\ \omega_p^{rs}(t), & t + \omega_p^{rs}(t) \in \left[t_a^* - \Delta, t_a^* + \Delta\right] \quad \text{punctual arrival} \\ \alpha\omega_p^{rs}(t) + \gamma\left[t + \omega_p^{rs}(t) - t_a^* - \Delta\right], & t + \omega_p^{rs}(t) > t_a^* + \Delta \quad \text{late arrival} \end{cases} \tag{1}$$

where $\omega_p^{rs}(t)$ is the *actual* travel time for travelers using path p^{rs} and departing at time t. α, β and γ are scalars satisfying $\gamma > \alpha > \beta > 0$.

A dynamic user optimal condition is attained *if and only if for each O–D pair, the actual commute costs experienced by travelers are equal and minimal no matter when they depart and which route they take*. Mathematically, the optimal condition is expressed as:

$$\begin{cases} f_p^{rs}(t)\left[c_p^{rs}(t) - \pi^{rs}\right] = 0, \forall r,s,p,t \\ c_p^{rs}(t) \geq \pi^{rs}, \qquad\qquad \forall r,s,p,t \end{cases} \tag{2}$$

where π^{rs} is the DUE path travel cost for users who belong to O–D pair rs, $f_p^{rs}(t)$ denotes the number of vehicles using path p^{rs} and departing at time t. We have the following flow conservation condition

$$\sum_{p^{rs} \in K^{rs}} \int_0^T f_p^{rs}(t)dt = q^{rs}, \forall r,s \tag{3}$$

and the nonnegativity condition:

$$f_p^{rs}(t) \geq 0, \forall r,s,p,t \tag{4}$$

Because one often needs to solve DUE problem numerically, a discrete form is presented below where an assignment period is divided into m intervals with an identical length δ. The DUO problem (2)–(4) is then reformulated as a discrete variational inequality, DVI(**c**, Ω), as follows:

$$\text{Find } \widetilde{f}_p^{rs}(t) \in \Omega \text{ such that } \sum_r \sum_s \sum_p \sum_t \widetilde{c}_p^{rs}(t)\left(f_p^{rs}(t) - \widetilde{f}_p^{rs}(t) \geq 0, \forall f_p^{rs}(t) \in \Omega, \right. \tag{5}$$

$$\Omega = \left\{ f_p^{rs}(t) \in R^+ \middle| \sum_{p^{rs} \in K^{rs}} \sum_{t=1}^{m} f_p^{rs}(t) = q^{rs}, \quad f_p^{rs}(t) \geq 0, \ \ \forall r,s,t \right\} \tag{6}$$

In matrix notation, DVI(**c**, Ω) can be simplified as

$$\text{Find } \widetilde{\mathbf{f}} \in \Omega \text{ such that } \left\langle \mathbf{c}\left(\widetilde{\mathbf{f}}\right), \mathbf{f} - \widetilde{\mathbf{f}} \right\rangle \geq 0, \quad \forall \mathbf{f} \in \Omega \tag{7}$$

$$\Omega = \{\mathbf{f} \in R^{n+} |, \mathbf{Mf} = \mathbf{q}, \mathbf{f} \geq 0\} \tag{8}$$

where $\langle a, b \rangle = a^T b$ and $n = m \times \sum_r \sum_s |K^{rs}|$.

It has been proven (Wei et al. 1995) that an **f** solves DVI(**c**, Ω) if and only if it satisfies the optimal conditions (2)–(4). A well-known existence result about DVI(**c**, Ω) is given below:

Theorem 1 *A solution exists for* DVI(**c**, Ω) *if* $\Omega \in R^n$ *is closed, convex and* **c**: $\Omega \rightarrow R^n$ *is continuous on* Ω.

Proof Via either Brouwer's fixed point theorem or degree theory. See Section 2.2 in Facchinei and Pang (2003). ■

Given that Ω is a polyhedral (hence a closed convex set), the question about the existence of a solution relies on the continuity of **c** with respect to **f**, and that was independently established in Wei et al. (1995) and Huang and Lam (2002) for the exit-flow function model and for the point-queue model, respectively. However, the continuity of **c** remains an open question for more realistic but also more complicated network traffic flow models such as those considering link interactions and queue spillover. The discontinuity of **c** in the presence of queue spillover may contribute to the instability of equilibrium solutions, as noted in Daganzo (1998). While this problem is more pronounced for DUO problems with no departure time choice, our intuition tells us that departure time choice would regularize the problem and the continuity of **c** can be expected in most cases, we therefore assume in this paper that the continuity of **c** holds and leave the study of existence and uniqueness of solutions to further research.

3 Dynamic network loading

A mesoscopic dynamic network loading (DNL) model is used in this study to evaluate the commute cost $c_p^{rs}(t)$, given an inflow pattern $f_p^{rs}(t), \forall p, r, s, t$. Path travel times $\omega_p^{rs}(t)$ (hence $c_p^{rs}(t)$) are computed by making use of link cumulative curves. As DNL proceeds, the cumulative arrival/departure curves on each link are built by counting the number of vehicles arriving and departing the link during each discrete loading interval δ. Time-dependent link traversal times are then retrieved from the cumulative curves provided the first-in-first-out (FIFO) condition is ensured throughout the loading process. In turn, travel times of individuals who depart at any assignment interval and travel along any given path can be calculated by recursively applying link travel times. Specifically, the traversal time of any vehicle entering a link a at time t_0 can be determined by

$$\omega(t_0) = D_a^{-1}(A_a(t_0)) - t_0 \tag{9}$$

where $D_a(t)$ and $A_a(t)$ are departure and arrival curves of link a. Thus, the path travel time $\omega_p^{rs}(t)$ can be recursively calculated as follows:

$$\omega_p^{rs}(t) = D_{a_s}^{-1}\left(A_{a_s}\left(D_{a_{s-1}}^{-1}\left(A_{a_{s-1}}\ldots\left(D_{a_0}^{-1}(A_{a_0})\right)\right)\right)\right) - t \tag{10}$$

where path p^{rs} contains a link sequence $\{a_0, a_1,..., a_s\}$. We emphasize that FIFO is a prerequisite to calculate path travel times using Eq. (10). Finally, path travel times are defined for each departure (loading) interval and measured in the unit of loading intervals. Note that a loading interval is usually much smaller than the assignment interval based on which the travel demand table is discretized.

The loading process is decomposed into two separate stages: propagate flows on links and transmit them through nodes. Three different models for link traffic dynamics are considered in this study: the point–queue (P–Q) model, the spatial queue (S–Q) model, and the kinematic wave model, known as the LWR model (Lighthill and Whitham 1955, Richards 1956). In the P–Q model, vehicles always move along a link at free-flow speed until they arrive at the exit point, where they

form a vertical queue if the outflow rate they induce exceeds the maximum discharge rate (capacity flow) of the link. Since the P–Q model ignores the physical length of vehicles, it never predicts a queue spillback. A simple remedy is to block inflow whenever a link's storage capacity is reached,[2] which gives the name to the S–Q model. Although the S–Q model captures the growth of queues across links, it implicitly supposes that any queue is at jam density (Zhang and Nie 2005), an assumption that is obviously in discord with most existing observations. The propagation speed of a queue is better predicted in the LWR model by explicitly tracing shockwaves. However, the LWR model in general requires more complex numerical solution schemes (e.g., the cell transmission model of Daganzo (1994) than the P–Q and S–Q models. Readers are referred to Zhang and Nie (2005) for more details on the similarities and differences among the three models.

To present a general model that governs traffic flow through nodes, we first define the *demand (or supply) of a link at time t*, $d(t)$ (or $s(t)$), as the maximum number of vehicles that are allowed to leave (enter) the link at t. Consider a general road intersection with n incoming and m outgoing links. At time t, each incoming link I_i and outgoing link O_i are associated with a demand $d_i(t)$ and a supply $s_i(t)$, respectively. Further, we assume that the proportion of vehicles on link I_i heading for each outgoing link O_j is known, denoted as a_{ij}. The flow between any pair of incoming link i and outgoing link j, $f_{ij}(t)$, can be computed as follows.

Step 1: Compute virtual demands for each incoming link i;

$$\mathrm{vd}_i(t) = \min\left(d_i(t), \max\left(\frac{s_j(t)}{a_{ij}(t)}, \forall j\right)\right) \tag{11}$$

Step 2: Compute virtual supply for each outgoing link i;

$$\mathrm{vs}_i(t) = \min\left(s_i(t), \sum_{j=1}^{m} a_{ij}(t) d_j(t)\right) \tag{12}$$

Step 3: Compute $f_{ij}(t)$ as follows:

$$f_{ij}(t) = \min\left(vd_i(t)a_{ij}(t), \mathrm{vs}_j(t)\frac{\mathrm{vd}_i(t)a_{ij}(t)}{\sum_{k=1}^{n} \mathrm{vd}_k a_{kj}(t)}\right) \tag{13}$$

Readers can verify that the merge and diverge models used in Daganzo (1995) and Jin and Zhang (2003) are special cases of this general model. A formal proof can be found in Chapter 5, Nie (2006). Note that the calculation of supply depends on link traffic dynamics. In our DNL model, the supply of a P–Q link always equals to infinity, while that of a S–Q link is the minimum of capacity flow and available holding space. The determination of supply in a LWR link is a bit more complex and not presented here to save space (See Daganzo 1995 for details).

[2] That is, the maximum number of vehicles a link can store under a given average vehicle length.

Link and node models are bound together by introducing two "buffers" into links. A link is thus divided into three parts: a core section which implements flow dynamics, an incoming and an outgoing flow buffer which handle the flow exchange between the link and the node. An incoming flow buffer is an imaginary facility in the sense that vehicles do not really spend time in it. It just temporarily stores incoming vehicles at the current loading interval. Conversely an outgoing flow buffer simulates road ends where vehicles heading for different directions are classified and queued in divided lanes. During each loading interval, nodes first transmit flows from outgoing buffers of upstream links into the downstream incoming buffers, using Eqs. (11)–(13). Then link models set forth to receive flows from their incoming buffers and send flows into their outgoing buffers. Obviously, the framework imposes no restriction on how nodes should compute the outgoing and incoming flows, or how links should transfer flows from its head to end. Hence any node/link model that defines the above operations can be adopted.

A notable feature of our DNL is the capability of tracking individual vehicular quanta. Each vehicular quantum is an indivisible flow element independently treated in the loading process like a vehicle in microscopic simulation. Among other advantages, tracing a vehicular quantum greatly reduces the complexity of ensuring FIFO. Obviously a queue structure is sufficient to preserve the order of vehicular quanta on links. Although the use of vehicular quanta makes it impossible to continuously represent traffic flow (note that any flow can only be measured in the unit of vehicular quantum), the introduced errors can always be controlled by reducing the size of the quantum.

Now we are ready to summarize the dynamic network loading procedure as follows:

Step 0: Initialize. Generate an empty network and let $t=0$.
Step 1: If $t \leq T$, for each $r \in R$, $s \in S$, create vehicular quanta from $q^{rs}(t)$ and insert them into the incoming flow buffers of the outgoing link of r.
Step 2: For each node $i \in N$, delete quanta from outgoing buffers of upstream links and insert them into the incoming buffers of downstream links, based on Eqs. (11)–(13).
Step 3: For each link $a \in A$, if a is connected with a destination, delete vehicular quanta from its incoming buffer; otherwise, move quanta from its incoming buffer to its outgoing buffer according to the specified traffic flow dynamics.
Step 4: If network is empty and $t > T$, stop; otherwise $t = t + \delta t$, go to Step 1.

4 Solution algorithms

We first present a few important analytical results based on which most of the studied algorithms are developed.

4.1 Merit-function transformation and other results

It is well known that DVI($\mathbf{c}$, Ω) cannot be transformed into a conventional mathematical program because the Jacobian of $\mathbf{c}$ is unlikely to be symmetric (note

that commuters' travel costs are only affected by those departing earlier than them, but not vice-versa). An alternative approach is to introduce merit (or gap) functions:

$$\text{Generic merit function(GMF)}\theta(\mathbf{f}) := \min_{g\in\Omega}\langle \mathbf{c}(\mathbf{f}), \mathbf{g}-\mathbf{f}\rangle \tag{14}$$

$$\text{Regularized merit function(RMF)}\theta_\tau(\mathbf{f}) := \min_{g\in\Omega}\langle \mathbf{c}(\mathbf{f}), \mathbf{g}-\mathbf{f}\rangle + \frac{1}{2\tau}\langle \mathbf{g}-\mathbf{f}, \mathbf{g}-\mathbf{f}\rangle \tag{15}$$

We have the following equivalence results.

Proposition 1 **f*** *is a solution to* DVI(**c**, Ω) *if and only if* **f*** *solves the maximization program* $\max_{f\in\Omega}\theta(f)$ *and* θ(**f***)=*0 is attained at the optimum*.

Proof If **f*** is a solution to DVI(**c**, Ω), then $\left\langle c\left(\mathbf{f}^*\right), \mathbf{g}-\mathbf{f}^*\right\rangle \geq 0$, so $\theta\left(\mathbf{f}^*\right) \geq 0$. On the other hand, by definition, $\theta\left(\mathbf{f}^*\right) \leq \left\langle c\left(\mathbf{f}^*\right), \mathbf{f}^*-\mathbf{f}^*\right\rangle = 0$. Consequently, θ(**f***) has to be zero and obviously this is the maximum possible value of θ(**f***). Thus, **f*** maximizes θ on Ω.

Conversely, if **f*** maximize θ on Ω, according to the definition we know that $\theta\left(\mathbf{f}^*\right) \leq 0$. And this "zero" is attained when taking minimum for $\left\langle c\left(\mathbf{f}^*\right), \mathbf{g}-\mathbf{f}^*\right\rangle, \forall g \in \Omega$, That is to say, $\left\langle c\left(\mathbf{f}^*\right), \mathbf{g}-\mathbf{f}^*\right\rangle \geq -, \forall g \in \Omega \mathbf{f}^*$ is a solution to DVI(**c**, Ω). ■

Note that θ(**f**) is not differentiable in general, while $\theta_\tau(\mathbf{f})$ is differentiable provided that **c** is continuously differentiable on Ω. However, the continuity of **c** in our DUO problem remains an unresolved issue, let alone its differentiability. Therefore, efficient algorithms would have to depend on "derivative-free" approaches. The following theorem lays a foundation for such algorithms

Theorem 2 *For every* **f**∈Ω, *the following three statements are valid*.

1. *The optimal solution to the minimum program that defines* $\theta_\tau(2f)$,$\widetilde{\mathbf{g}}_\tau(\mathbf{f})$, *is the projection of* **f**-τ**c**(**f**) *onto* Ω, *i.e.*,

$$\widetilde{\mathbf{g}}_\tau(\mathbf{f}) = \Pi_\Omega(\mathbf{f}-\tau\mathbf{c}(\mathbf{f})) \tag{16}$$

2. *If* **c**(**f**) *is Lipschitz continuous and strongly monotone on* Ω, *then* $\widetilde{\mathbf{g}}\tau(\mathbf{f})-\mathbf{f}$ *is a strict ascent direction for* $\theta_\tau(\mathbf{f})$ *at* **f**, *whenever* **f** *is not a solution to* DVI(**c**, Ω).
3. *If* **c**(**f**) *is Lipschitz continuous and monotone on K, either* $\widetilde{\mathbf{g}}\tau(\mathbf{f})-\mathbf{f}$ *is a strict ascent direction for* $\theta_\tau(\mathbf{f})$ *at* **f**, *or* $\theta_\tau(\mathbf{f}) = \frac{1}{2\tau}\langle \mathbf{g}-\mathbf{f}, \mathbf{g}-\mathbf{f}\rangle$.

Proof The first statement follows from Theorem 10.2.3 in Facchinei and Pang (2003) , while the second and third statements follow from Propositions 1 and 2 in Zhu and Marcotte (1993), respectively. ■

The projection $\Pi_\Omega(\mathbf{f}-\tau\mathbf{c}(\mathbf{f}))$ provides not only a feasible ascent direction, but also a means to verify optimality, as shown in Theorem 3.

Theorem 3 $\mathbf{f}$ *solves* DVI($\mathbf{c}$, Ω) *if and only if* $\mathbf{f} = \Pi_{\Omega}(\mathbf{f} - \tau\mathbf{c}(\mathbf{f}))$.

Proof Follows from Nagurney (1993).

This result plays a fundamental role in designing projection-type VI algorithms, of which some will be explored in Section 4.2.

We now return to the generic merit function $\theta(\mathbf{f})$ to explore some of its interesting properties that would lead to another feasible direction algorithm.

Proposition 2 $\widetilde{\mathbf{g}}(\mathbf{f})$ *solves the minimum program that defines* $\theta(\mathbf{f})$ Eq. (14) *for a given* $\mathbf{f}$ *if and only if* $\widetilde{\mathbf{g}}(\mathbf{f})$ *is an all-or-nothing assignment on time-dependent minimum-cost paths for each O–D pair.*

The proof of this is obvious and not presented here to save space. Existing heuristics (e.g., the method of successive averages) for the DUO problem often assume that $\widetilde{\mathbf{g}}(\mathbf{f})$ is a feasible direction along which $\theta(\mathbf{f})$ could be improved. However, it seems difficult to prove that $\widetilde{\mathbf{g}}(\mathbf{f})$ really provides an ascent direction. The following proposition offers an alternative.

Proposition 3 *Let* $\widetilde{\mathbf{g}}(\mathbf{f})$ *solve the minimum program defined in* Eq. (14) *for a given* $\mathbf{f}$. *If* $\mathbf{c}$ *is strictly monotone, and* $\mathbf{y} = \widetilde{\mathbf{g}}((1+\lambda)\mathbf{f} + \lambda\mathbf{y}), \lambda \in (0,1)$, *then* $\mathbf{y} - \mathbf{f}$ *is a strict ascent direction for* $\theta(\mathbf{f})$ *at* $\mathbf{f}$.

A proof of this result is provided in Nie and Zhang 2005 thus not repeated here. We shall revisit this result in Section 4.2 to develop a line search strategy that avoids solving $\mathbf{y}$ explicitly.

4.2 Algorithms

The focus is on the feasible direction algorithms, but for completeness and comparison purpose a brief review of heuristic and projection-type algorithms is also given. Besides algorithms included in our study, two recently developed VI algorithms are worthy of attention. Han and Lo (2002) proposed a simple projection method that reduces the overhead of performing projection. The method, known as the modified alternating direction (MAD) algorithm, simplifies the constraint structure by introducing a vector of Lagrange multiplier (associated with the flow conservation constraints) into solution variables. An application of this algorithm in solving a DUE problem was reported in Lo and Szeto (2002). Note that although MAD simplifies projection, it notably enlarges the solution space which may negatively impact the overall convergence. Another algorithm, called the projection and contraction (PC) method, was originally proposed by He (1997) to solve general continuous and monotone VI problems. He's original method includes a line search procedure to find optimal step size. Chen et al. (2001) later designed a self-adaptive rule to replace the line search in the PC method. They use the self-adaptive PC method to solve the static traffic assignment problems. The application of both PC methods for solving the DUE problem is not found in the literature.

4.2.1 Heuristic algorithms

Heuristic algorithms always try to shift flows from time-dependent non-minimum-cost path onto the minimum-cost one. The simplest flow-shifting strategy is the so-called method of successive averages (MSA).

Algorithm MSA

Step 1: Set $\lambda=1/k$ where k is the iteration index, obtain the new solution by a

$$\mathbf{f}^{k+1} = (1-\lambda)\mathbf{f}^k + \lambda\widetilde{g}\left(\mathbf{f}^k\right) \tag{17}$$

where $\widetilde{g}\left(\mathbf{f}^k\right)$ is defined in Proposition 2.

MSA is known to converge to the optimal solution in the static traffic assignment problem, as shown in Sheffi (1985). Although existing experiments (Tong and Wong 2000) indicate that MSA can produce a solution that satisfactorily fulfill the dynamic user equilibrium condition, its convergence in the DUO context has never been rigorously justified.

4.2.2 Projection algorithms

The simplest version of the projection algorithm involves a single Euclidean projection in each iteration, i.e., obtaining a new solution by solving

$$\mathbf{f}^{k+1} = \Pi_\Omega\left(\mathbf{f}^k - \tau c\left(\mathbf{f}^k\right)\right) \tag{18}$$

To ensure the convergence, the mapping $\mathbf{c}$ has to be Lipschitz continuous (with L), strongly monotone (with μ), and $\tau < 2\mu/L^2$ (Theorem 12.1.2, Facchinei and Pang 2003). A simple revision of the above algorithm, called extra projection, can reduce the convergence requirement to Lipschitz continuity plus pseudo monotonicity. The extra projection algorithm performs projection twice in each iteration as below.

$$\mathbf{y}^k = \Pi_\Omega\left(\mathbf{f}^k - \tau\mathbf{c}\left(\mathbf{f}^k\right)\right) \tag{19}$$

$$\mathbf{f}^{k+1} = \Pi_\Omega\left(\mathbf{f}^k - \tau\mathbf{c}\left(\mathbf{y}^k\right)\right) \tag{20}$$

Note that the extra projection algorithm (EPA) still requires the estimate of Lipschitz constant L because its convergence is ensured only if τ is less than L. It is difficult to predetermine L in the DUO problems. However, a practical resolution would be starting from an initial guess of τ then reducing it whenever a symptom of the violation of convergence criterion appears (e.g., a satisfying improvement of merit function value cannot be achieved in an iteration). This idea is exposited as follows

Algorithm EPA

Step 1: Obtain new solution $\mathbf{f}^{k+1}$ using Eqs. (19) and (20). Evaluate merit function $\theta(\mathbf{f}^{k+1})$. If

$$\theta\left(\mathbf{f}^{k+1}\right) - \theta\left(\mathbf{f}^k\right) < 0 \quad \text{and} \quad \left|\theta\left(\mathbf{f}^{k+1}\right) - \theta\left(\mathbf{f}^k\right)\right| / \left|\theta\left(\mathbf{f}^k\right)\right| > \varepsilon, \tag{21}$$

set $\tau = \tau \times \sigma$, where ε and σ are positive scalars between 0 and 1.

4.2.3 Feasible direction algorithms

We first review a feasible direction algorithm proposed by Zhu and Marcotte (1993), which is based on Theorem 2. We name it the projection feasible direction algorithm (PFD) since the algorithm maximizes RMF $\theta_\tau(\mathbf{f})$ along the direction defined by a projection. The convergence of PFD requires that the mapping $\mathbf{c}$ is monotone and Lipschitz continuous. Nevertheless, unlike algorithm EPA an estimate of Lipschitz constant is not necessary.

Algorithm PFD

Step 1: Compute $\mathbf{y}^k = \Pi_\Omega\left(\mathbf{f}^k \tau \mathbf{c}\left(\mathbf{f}^k\right)\right)$. Evaluate $\theta_\tau\left(\mathbf{f}^k\right)$ at $\mathbf{y}^k$ using Eq. (15) (note that this requires another projection at $\mathbf{y}^k$)

Step2: If

$$\theta_\tau\left(\mathbf{f}^k\right) \geq \frac{1}{2\tau(1-\rho)}\left\|\mathbf{y}^k - \mathbf{f}^k\right\|^2, \tag{22}$$

where ρ is a positive scalar smaller than 1, set $\tau = \tau/\sigma$, $\sigma \in (0,1)$, then update $\mathbf{f}^{k+1}=\mathbf{f}^k$; otherwise, find the largest λ^k such that $\theta_\tau\left(\mathbf{z}^k\right) > \theta_\tau\left(\mathbf{f}^k\right)$, where $\mathbf{z}^k = (1-\lambda)\,\mathbf{f}^k+\mathbf{y}^k$, then update $\mathbf{f}^{k+1}=\mathbf{z}^k$.

The Armijo rule is used for line search in Step 2. In order to reduce the number of line searches conducted in each iteration, the choice of the initial step size is based on the step size used in the last iteration.

An Armijo line search for PFD

Step 0: Set $\lambda^k=\min\{1.0,\ 2\times\lambda^{k-1}\}$

Step 1: Compute $\mathbf{z}^k=(1-\lambda)\,\mathbf{f}^k+\lambda\mathbf{y}^k$. If $\theta_\tau(\mathbf{z}^k)>\theta_\tau(\mathbf{f}^k)$ is satisfied, stop; otherwise set $\lambda^k=0.5\times\lambda^k$, repeat Step 1.

We now present an alternative feasible direction algorithm (AFD) that minimizes the generic instead of the regularized merit function, based on Proposition 3. In order to obtain the feasible direction defined in Proposition 3, one has to solve

$$\mathbf{y} = \widetilde{g}((1-\lambda)\mathbf{f} + \lambda\mathbf{y}), \tag{23}$$

for $\mathbf{y}$, where $\mathbf{f}$ is a constant vector and $\widetilde{g}$ $(\mathbf{f})$ minimizes the mathematical program that defines θ $(\mathbf{f})$. If $\widetilde{g}(\cdot)$ is a contraction mapping, according to Banach fixed-point theorem, we have (1) the mapping $\widetilde{g}(\cdot)$ has a unique fixed point in Ω, and (2) for any starting point $\mathbf{y}_0$, the following contraction iterate

$$\mathbf{y}^{k+1} = \widetilde{g}\left((1-\lambda)\mathbf{f} + \lambda\mathbf{y}^k\right) \tag{24}$$

will generate a sequence $\{\mathbf{y}^k\}$ converging to the fixed point (Theorem 2.1.21, Facchinei and Pang 2003). However, it is difficult to show that $\widetilde{g}(\cdot)$ is indeed a contraction. Even if it is, computing the ascent direction exactly may be computationally intensive thereby not suitable for practical purposes. We thus introduce a line search procedure that attempts to move along this ascent direction approximately.

Algorithm AFD

Step 1: Compute $\mathbf{y}^k = \widetilde{\mathbf{g}}(\mathbf{f}^k)$ where $\widetilde{\mathbf{g}} = (\mathbf{f}^k)$ is defined in Proposition 2. Evaluate θ ($\mathbf{f}^k$) at $\mathbf{y}^k$ using Eq. (9). Set λ^k=min{1.0, $2\times\lambda^{k-1}$}.

Step 2: Compute $\mathbf{z}^k$=(1−λ) $\mathbf{f}^k$+λ$\mathbf{y}^k$. If θ ($\mathbf{z}^k$)>θ ($\mathbf{f}^k$), $\mathbf{f}^{k+1}$=$\mathbf{z}^k$; otherwise, set $\mathbf{y}^k = \widetilde{\mathbf{g}}(\mathbf{z}^k)$ and λ^k=0.5×λ^k , repeat Step 2.

AFD updates via Eq. (24) its search direction $\mathbf{y}^k$ after each line search in the hope of getting a better approximation to the solution of Eq. (23). Certainly this would be true only if the mapping $\widetilde{g}(\cdot)$ is a contraction.

Finally, we present a heuristic feasible direction (HFD) algorithm that employs the solution from an extra projection as a search direction. As noted, the extra projection algorithm requires weaker conditions for convergence, thus we expect that it might provide a better ascent direction. To reduce computational cost, the value of GMF (general merit function) instead of RMF (regularized merit function) is used to guide the line search procedure. The major steps of the HFD algorithm are summarized below:

Algorithm HFD:

Step 1: Obtain a new solution $\mathbf{y}^k$ using Eqs. (19) and (20). Evaluate θ ($\mathbf{f}^k$) at $\mathbf{y}^k$ using Eq. (14). Set λ^k=min{1.0, $2\times\lambda^{k-1}$}.

Step 2: Compute $\mathbf{z}^k$=(1−λ) $\mathbf{f}^k$+λ$\mathbf{y}^k$. If θ ($\mathbf{z}^k$)>θ ($\mathbf{f}^k$), $\mathbf{f}^{k+1}$=$\mathbf{z}^k$; otherwise, set $\mathbf{y}^k = \Pi_\Omega(\mathbf{f}^k - \tau\mathbf{c}(\mathbf{z}^k))$ and λ^k=0.5×λ^k, repeat Step 2.

4.2.4 Other implementation issues

To avoid enumerating all paths from the beginning (which is prohibitive for large problems), a column generation scheme is used to iteratively build the optimal path set. This requires searching for time-dependent minimum-cost paths (TDMCP) based on the current loading results. In this study, we adopted a revised version of the decreasing order of time (DOT) algorithm (Chabini 1998) for column generation. During each iteration, TDMCP is called to generate shortest paths for each O–D pair and each departure time interval. The new paths will be added into the path set of a corresponding O–D pair. Each departure time interval stores its own assignment elements (each element contains a path pointer and the flow assigned to it). Although paths will never be deleted from the path set, an assignment element will be removed from a departure time interval if the assigned flow equals zero.

Performing the aforementioned projection involves solving a quadratic programming problem. Given the special structure of the feasible set (it is a simplex and decomposable), this problem can be efficiently solved using the reduced gradient algorithm. Note that one quadratic programming problem needs to be solved for each O–D pair.

In all experiments, an initial solution $\mathbf{f}_0$ is chosen such that demands of each O–D pair q^{rs} are assigned to initial shortest paths (i.e., the shortest paths when network is empty), but evenly distributed to each assignment interval.

The value of $\theta(\mathbf{f})$ is a natural convergence measure. Note that the maximum of $\theta(\mathbf{f})$ is zero (Facchinei and Pang 2003), thus $|\theta(\mathbf{f})|$ can be used as a termination criterion. Specifically, an algorithm is terminated when either the iteration number reaches the maximum allowed value $k_{\max}$, or the following criterion is satisfied:

$$\overline{\theta}(\mathbf{f}) = \frac{\theta(\mathbf{f})}{\langle \widetilde{\mathbf{g}}(\mathbf{f}), c(\mathbf{f}) \rangle} \leq \varepsilon, \tag{25}$$

where $\theta(\mathbf{f}) = \min_{g \in \Omega} \langle \mathbf{c}(\mathbf{f}), \mathbf{g} - \mathbf{f} \rangle = \langle \mathbf{c}(\mathbf{f}), \widetilde{\mathbf{g}}(f) - \mathbf{f} \rangle$. Note that $\overline{\theta}(\mathbf{f})$ equals to the duality gap used in Tong and Wong (2000).

For the three feasible direction algorithms (PFD, AFD and HFD), obtaining an optimal step size is not always possible since the obtained feasible directions are not necessarily ascent directions. We thus need to avoid the potential "breakdown" caused by a non-ascent direction. In our implementation, a line search procedure is forced to terminate as long as the current step size is less than the minimum allowed value $\lambda_{\min}$, which is then used to obtain a new solution.

5 Numerical studies on two sample networks

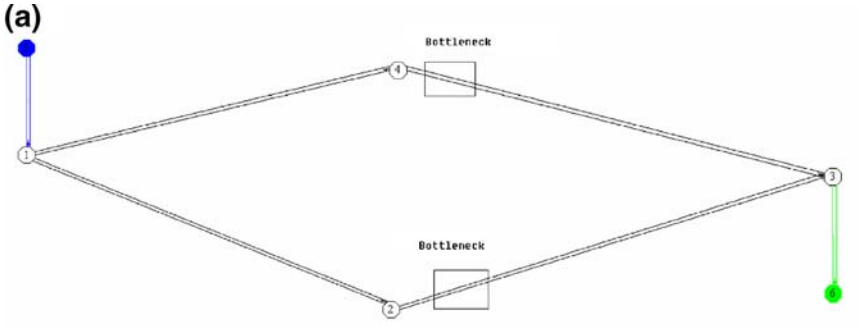
The five DUO algorithms as well as the presented dynamic network loading model were coded in MS-VC++ and tested on a Windows-XP server (with 3.06 GHz CPU and 2 Gb memory) for two scenarios. The first scenario considers a network with a single O–D pair connected by two parallel paths. It is set up to illustrate that the proposed solution framework can produce numerical solutions comparable to the analytical solution. A more complex network is used in the second scenario to explore the impact of various traffic dynamics on equilibrium solutions. The performance of different algorithms was compared in both scenarios.

In all experiments, we set α=6.4 ($/h), β=3.9 ($/h) and γ=15.21($/h), based on the empirical study of Small (1982). The starting point of the assignment horizon is always set as 6:00 AM for convenience. Further, the length of an assignment interval δ is 1 min, while the size of vehicular quantum is 0.5.

5.1 Scenario I

As shown in Fig. 1, O–D pair 5–6 is connected by paths 1 and 2. The capacities of links 4–3 and 2–3 (modeled by S–Q dynamics) is set as 2,000 and 1,000 vph respectively, lower than those of their upstream links (4,000 vph). As such, the entrances of links 4–3 and 2–3 serve as bottlenecks. To make sure queues only appear before node 4 or 2, links 1–4 and 1–2 are modeled by the P–Q dynamics (note that the supply of P–Q model is always treated as infinity). Other parameters are provided in Fig. 1.

(a)



Path 1: 5-1-4-3-6 (node sequence)	
Path 2: 5-1-2-3-6 (node sequence	
Free flow travel time on path 1	12 min
Free flow travel time on path 2	18 min
T	7:00 am
t*	6:48 am
Δ	6 min
q^{56}	2000
Loading interval δt	30 seconds

(b)

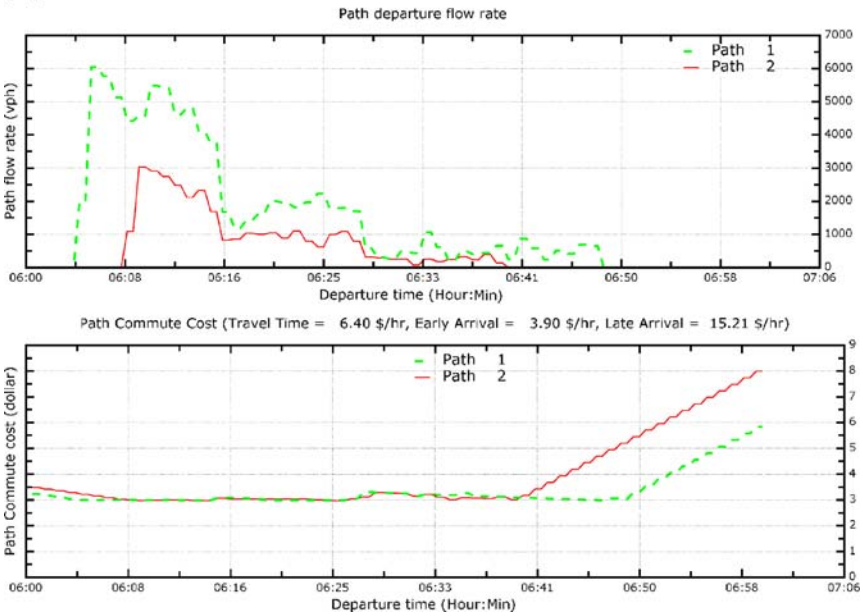


Fig. 1 Network topology and DUO solution for scenario I. (**a**) Network topology. (**b**) DUO solution

Table 1 Parameters, iteration numbers and CPU times of various algorithms in both scenarios

	Scenarios	k_{max}	λ_{min}	σ	τ	ρ	ε	k_c	CPU time
MSA	I	4,000	0.001	N/A	N/A	N/A	0.025	1,747	28.9
	II	1,000	0.001	N/A	N/A	N/A	0.05	235	15.3
EPR	I	4,000	0.0005	0.9	0.5	N/A	0.025	k_{max}	103.3
	II	1,000	0.0005	0.9	0.5	N/A	0.05	65	5.3
PFD	I	4,000	0.001	0.8	0.5	0.95	0.025	k_{max}	123.9
	II	1,000	0.005	0.8	0.1	0.9	0.05	372	32.3
AFD	I	4,000	0.001	N/A	N/A	N/A	0.025	381	16.8
	II	1,000	0.001	N/A	N/A	N/A	0.05	166	18.1
HFD	I	4,000	0.05	N/A	0.5	N/A	0.025	248	11.7
	I	1,000	0.0005	N/A	5	N/A	0.05	94	11.3

k_{max} the maximum allowed iteration number, λ_{min} the minimum allowed step size, σ Scalar as described in Algorithm EPR and PFD, τ projection factor, as defined in Eq. (18), ρ as defined in Eq. (22), ε the convergence criterion, as used in Eq. (25), k_c number of iterations spent when the convergence criterion (25) is satisfied, *CPU time* the seconds of CPU time consumed when the convergence criterion (25) is satisfied

Using the formulas developed in Arnott et al. (1990), we calculate optimal path departure flow rates as follows

$$f_1(t) = \begin{cases} 5,120, & 6.07 \le t < 6.24 \\ 2,000, & 6.24 \le t \le 6.44 \\ 592.3, & 6.44 < t \le 6.81 \\ 0, & \text{otherwise} \end{cases} \quad f_2(t) = \begin{cases} 2,560, & 6.14 \le t < 6.24 \\ 1,000, & 6.24 \le t \le 6.44 \\ 296.2, & 6.44 < t \le 6.67 \\ 0, & \text{otherwise} \end{cases}$$

The total flows assigned to paths 1 and 2 are 1,471 and 529 respectively, while the optimal commute cost is 2.92$.

Now we examine numerical solutions produced by the DUO algorithms. Figure 1 plots the curves of path flow rates and commute costs against departure times, which correspond to an equilibrium gap of 0.025. It is easy to verify from the figure that the DUO conditions are well satisfied. Although the numerical solution does not replicate the exact analytical results, it captures the duration of the rush hour and the transition between the formation and dissipation of queues rather accurately.

The number of iterations and CPU time that each algorithm spent to reach the required accuracy is reported in Table 1. Table 1 also lists the major parameters that may affect an algorithm's performance. The parameters are chosen such that the performance is optimized based on trial-and-error. Note that HFD consumes the least iterations and CPU time to reach the equilibrium gap of 0.025, followed by AFD and MSA. PFD and EPR did not converge after 4,000 iterations. A detailed comparison of convergence performance of the algorithms is reported in Fig. 2. For the sake of clarity, the results after 250 iterations or 10 s CPU time are ignored. For most algorithms, the drop of the equilibrium gap is rather sharp in the beginning, and then significantly flattens as the solution is getting closer to optimum. The EPR algorithm oscillates strongly in the early stage, probably due to that the value of τ is larger than the Lipschitz constant. Nevertheless, its convergence performance gets improved as

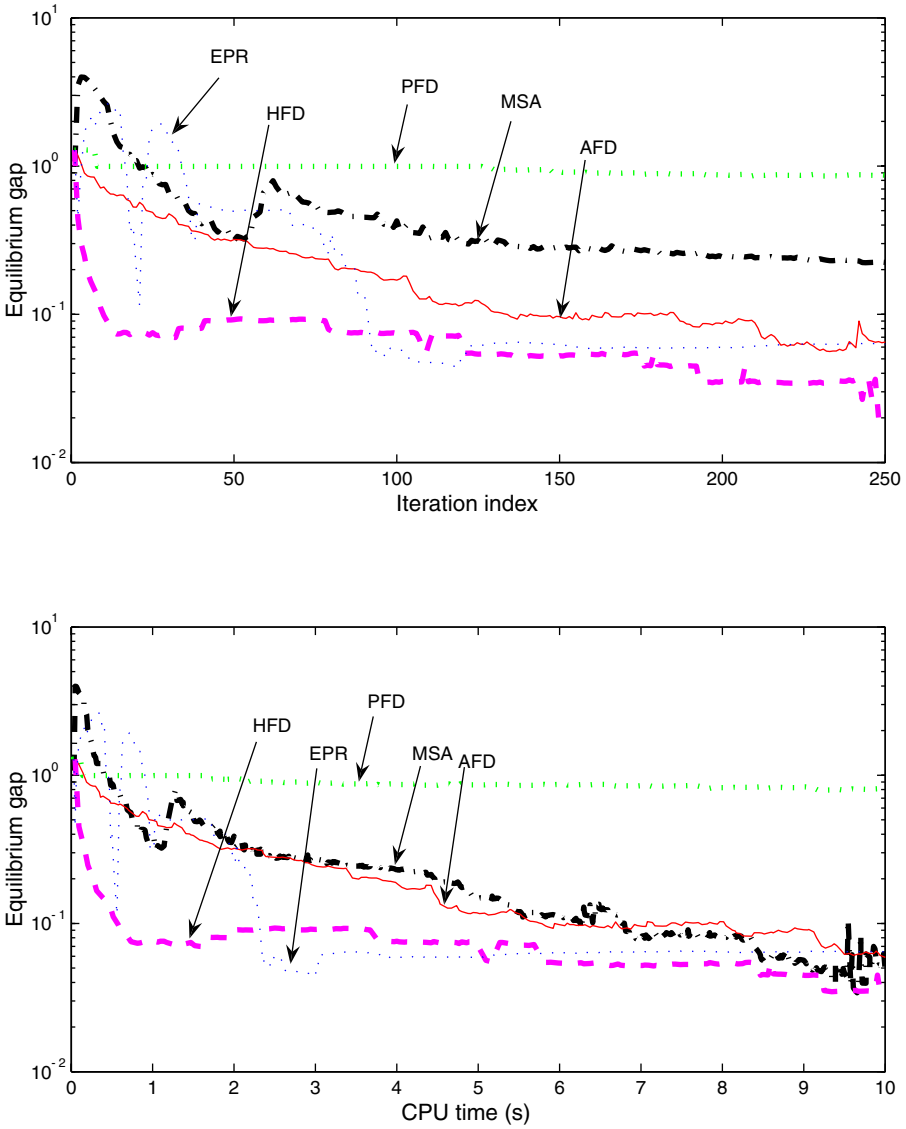


Fig. 2 Convergence curves of various algorithms for scenario I

τ becomes smaller. The convergence curves of feasible direction algorithms look smoother in general. This is expected since the embedded line search procedure helps avoid improper step sizes. Further, HFD and AFD consistently outperform MSA in terms of both iterations and total CPU time consumed, even if they need to conduct a line search. Finally, the convergence of PFD is relatively poor in this example. It is possibly due to violations of the assumption that ensure convergence and/or that the related parameters are not appropriately selected.

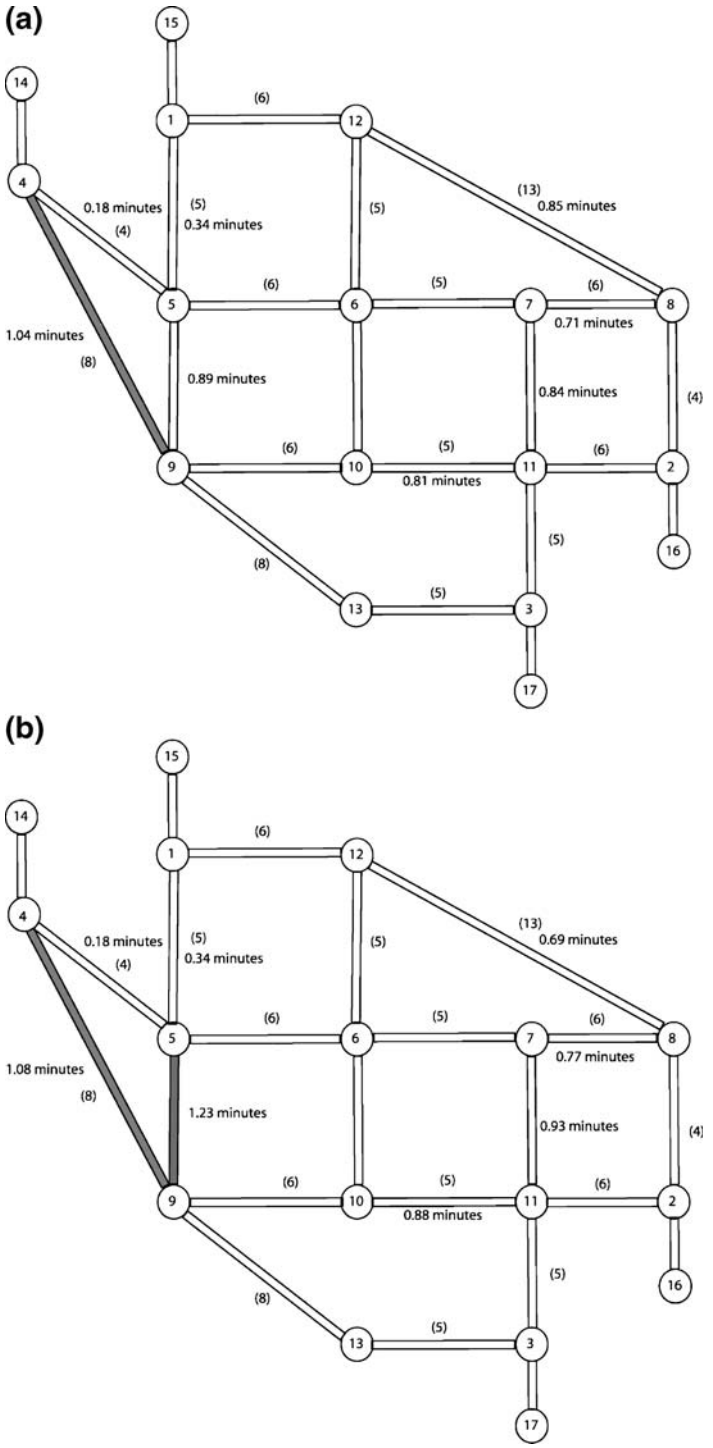


Fig. 3 Average equilibrium link travel delays corresponding to different traffic dynamics. (**a**) LWR case, (**b**) SQ case. Numbers in the parenthesis are link free flow travel times in the unit of loading interval

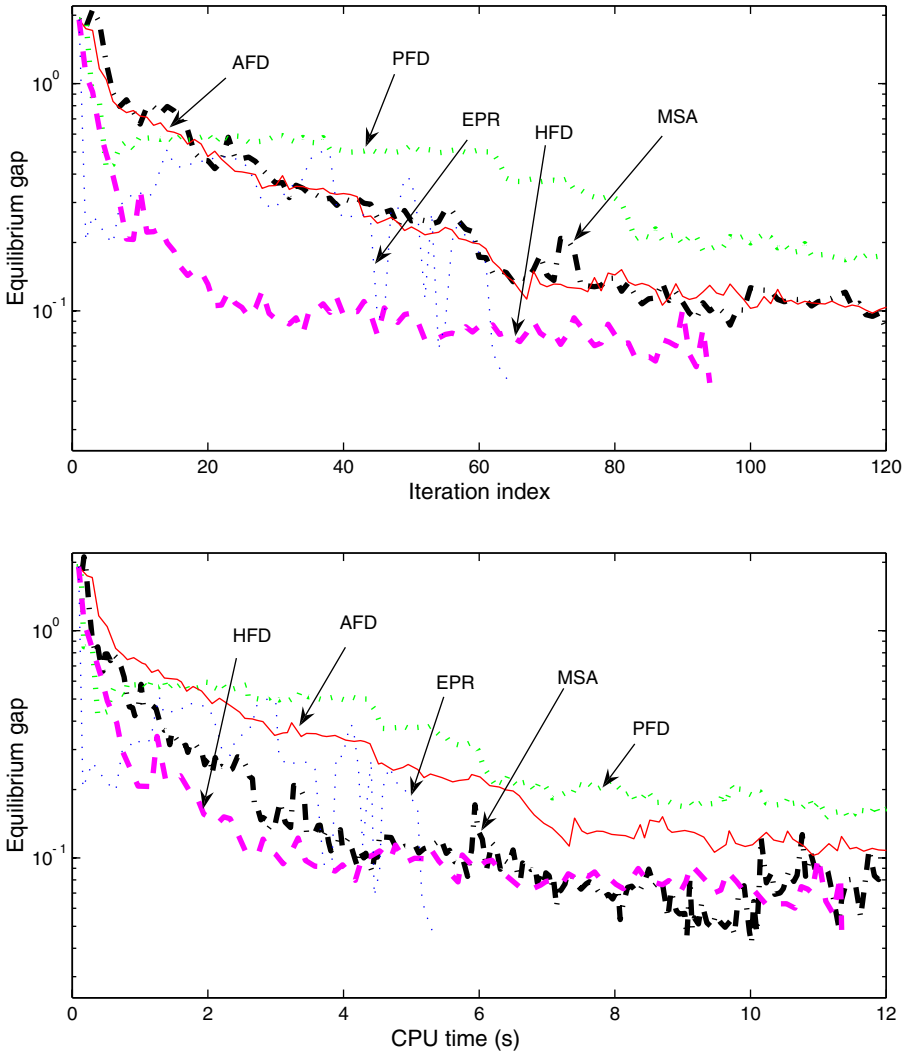


Fig. 4 Convergence curves of various algorithms for scenario II

5.2 Scenario II

In this scenario, we consider a network similar to that of Nguyen and Dupius (1984), which is composed of 17 nodes, 23 links and 4 O–D pairs (Fig. 3). Links 14–4, 15–1, 3–17 and 2–16 are dummy links introduced to facilitate dynamic network loading. The LWR model with a triangle fundamental diagram is used to describe traffic dynamics on all non-dummy links except on links 1–5, 4–5 and 4–9. For these three links we adopt the P–Q model to avoid queues on dummy links. Unless otherwise specified, the free flow speed, flow capacity, holding capacity and number of lanes on all non-dummy links are 30 mph, 1,800 vphpl, 200 vpmpl and 3, respectively. To "create" bottlenecks, the number of lanes on links 9–13, 11–3, and 8–2 is set as 2, 1 and 2, respectively. In order to make it easier to observe spillback, we also reduce

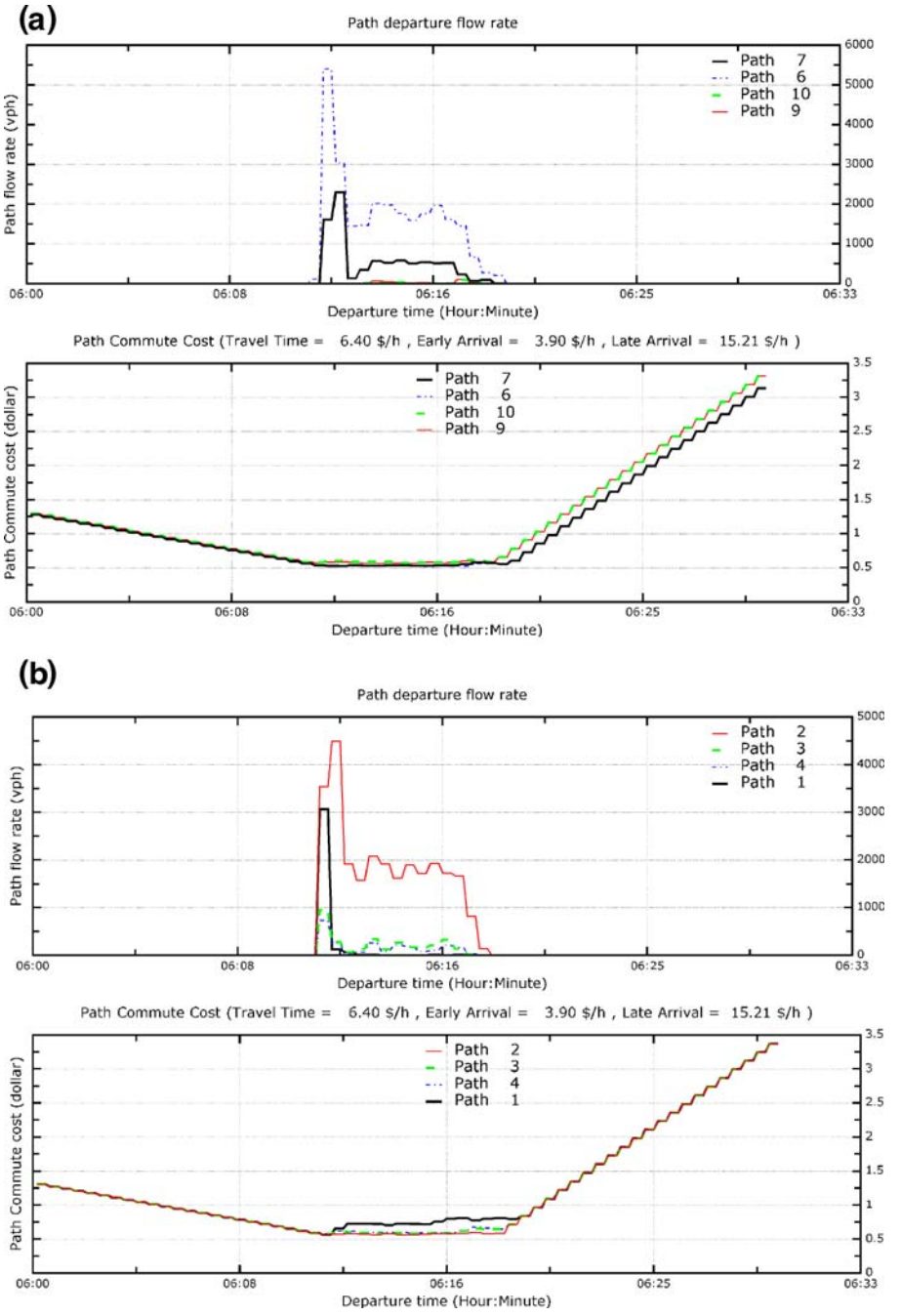


Fig. 5 DUO solutions for scenario II. (**a**) O–D pair 14–17, (**b**) O–D pair 14–16

Table 2 Path flow pattern comparison

OD	Path ID	Path Topology	Total flows in LWR case	Total flows in SQ case
14–17	6	14–4–9–13–3	223.36	236.62
	7	14–4–5–9–13–3	71.41	30.29
	8	14–4–9–10–11–3	0.00	0.00
	9	14–4–5–6–7–11–3	3.07	18.80
	10	14–4–5–6–10–11–3	2.17	11.56
	11	14–4–5–9–10–11–3	0.00	0.00
14–16	1	14–4–9–10–11–2	27.75	3.19
	2	14–4–5–6–7–8–2	225.33	222.37
	3	14–4–5–6–7–11–2	26.67	40.60
	4	14–4–5–6–10–11–2	20.25	33.83
	5	14–4–5–9–10–11–2	0.0	0.00

the free flow speed on link 5–9 to 10 mph. We set the other parameters to be: T= 6:30 AM, Δ=5 min, t^*=6:20 AM, loading interval δt=10 s.

The convergence curves of the five algorithms are given in Fig. 4 (results after 120 iterations or 12 s CPU time are ignored). As reported in Table 1, EPR achieves the required accuracy with least iterations, followed by HFD, AFD and MSA. Roughly speaking, the relative performance of the algorithms is similar to that of the first scenario. However, the computational superiority of the two feasible direction algorithms, HFD and AFD, is weakened. Note that AFD is always slower than MSA, even if it needs slightly less iterations to achieve a same equilibrium gap. For one thing, the complexity of the problem structure might make it harder to fulfill the convergence requirements. As a result, the adopted feasible directions may deviate from a truly ascent direction more easily. Moreover, network loading is more time-consuming in this larger example, making line search relatively more expensive.

For brevity, only the optimal path flow patterns for O–D pair 14–16 and 14–17 are reported in Fig. 5. DUO conditions are approximately satisfied for both O–D pairs, namely, the path/s used at any departure time have roughly equal and minimum commute costs whereas any unused ones have higher commuter costs. Further, the rush hour congestion evolution in this example follows a similar pattern as depicted in Scenario I, consisting of distinct stages of queue formation, stagnation and dissipation. Average link travel delays are given in Fig. 3(a). As expected, upstream links of the designated bottlenecks subject to high average delay, implying that queues developed on those links during the rush hour. Particularly, the delays on links 1–5 and 4–5 are caused by queue spillback of link 5–9. The maximum number of queued vehicles on link 5–9 is 55 vehicles, less than its storage capacity of 75. As known, spillback can occur well before the holding space runs out because queues are not necessarily at jam density in the LWR model.

We proceed to investigate how queue spillback affects equilibrium solutions, by replacing the traffic dynamic model of link 5–9 with the S–Q model. Since the S–Q model always predicts queue spillback later than the LWR model (Zhang and Nie 2005), it is expected that spillback may not happen in this case.

Figure 3(b) shows the average link travel delays at equilibrium. An immediate observation is that there is no delay on links 1–5 and 4–5 and the delay on link 5–9 increases by 40%. This is because congestion on link 5–9 did not travel across node

5. A detailed examination shows that the maximum queued vehicles on links 5–9 is about 65, higher than the LWR case but not high enough to trigger a spillback. Table 2 compares the path usage at equilibrium for O–D pairs 14–16 and 14–17. For O–D pair 14–17, the total flow using path 7 drops more than 50% in the S–Q case. This is due to that the increased queue length on link 5–9 makes the path less attractive to travelers. For O–D pairs 14–16, the usage of path 3 and 4 is increased more than 50% in the S–Q case. These two paths attracted more traffic because link 4–5 is free of interference with congestion on link 5–9.

To summarize, this example demonstrates that equilibrium solutions vary substantially as the propagation speed of queues on a single link is modeled differently. When one chooses to use certain simplified link traffic flow models one need to bear in mind the type and extent of approximation errors that one may encounter in the process.

6 Conclusions

We have studied the dynamic user optimal (DUO) assignment problem considering simultaneous route and departure time choice, formulated as a discrete variational inequality (DVI). To evaluate dynamic path travel costs, a mesoscopic dynamic network loading (DNL) model is developed that considers interactions between links and the spatial allocation of queues. Various solution techniques for the DUO problem, particularly those based on feasible directions and line searches, are examined in this paper.

Our findings from the numerical experiments are summarized as follows:

1. Numerical solutions in both scenarios precisely follow the DUO principle. The pattern of queue formation, stagnation and dissipation obtained from numerical results well matches the analytical results.
2. Introducing line searches provides relatively faster and more stable convergence, compared with the popular heuristic method of successive averages (MSA). When appropriately implemented, the feasible direction algorithms can outperform MSA in terms of computational overhead.
3. The dynamic user equilibrium traffic pattern is sensitive to the spatial distribution of queues. To model queuing phenomena adequately would require predicting the propagation of queues through links in a realistic way. This in turn calls for models capable of tracing the moving speed of queue ends (i.e., shockwave speed).

A fundamental issue worthy of further investigation is the analytical properties of the path cost function **c**. For one thing, the continuity of **c** in the presence of link interactions remains an unresolved problem, thus raising the question on the existence of DUO solutions. Moreover, better understanding the nature of **c** would help design more efficient DUO solution algorithms.

Acknowledgement The authors would like to thank the anonymous referees for their comments andsuggestions to improve the paper. This research is supported in part by a research grant from Caltrans. The authors are solely responsible for the content of this paper.

References

Arnott R, de Palma A, Lindsey R (1990) Departure time and route choice for routes in parallel. Transp Res 24B:209–228

Beckmann M, McGuire CB, Winsten CB (1956) Studies in the economics of transportation. Yale University Press, New Haven, CT

Chabini I (1998) Discrete dynamic shortest path problems in transportation applications. Transp Res Rec 1645:170–175

Chen H-K, Hsueh C-F (1998) A model and an algorithm for the dynamic user-optimal route choice problem. Transp Res 32B:219–234

Chen A, Lo H, Yang H (2001) A self-adaptive projection and contraction algorithm for the traffic assignment problem with path-specific costs. Eur J Oper Res 135:27–41

Daganzo CF (1994) The cell transmission model: a dynamic representation of highway traffic consistent with the hydrodynamic theory. Transp Res 28B:269–287

Daganzo CF (1995) The cell transmission model, Part II: network traffic. Transp Res 29B:79–93

Daganzo CF (1998) Queue spillovers in transportation networks with a route choice. Transp Sci 32:3–11

Facchinei F, Pang J-S (2003) Finite-dimensional variational inequalities and complementarity problems, vol. I, II. Springer, New York

Friesz TL, Mookherjee R (2006) Solving the dynamic network user equilibrium problem with state-dependent time shifts. Transp Res 40B:207–229

Friesz TL, Bernstein D, Smith TE, Tobin RL, Wei BW (1993) A variational inequality formulation of the dynamic network equilibrium problem. Operation Research 41:179–191

Friesz TL, Bernstein D, Suo Z, Tobin RL (2001) Dynamic network user equilibrium with state-dependent time lags. Networks and Spatial Economics 1:319–349

Han D, Lo H (2002) New alternating direction method for a class of nonlinear variational inequality problems. J Optim Theory Appl 112:549–560

He B (1997) A class of projection and contraction methods for monotone variational inequality. Appl Math Optim 35:69–76

Huang HJ, Lam WHK (2002) Modeling and solving dynamic user equilibrium route and departure time choice problem in network with queues. Transp Res 36B:253–273

Jin WL, Zhang HM (2003) On the distribution schemes for determining flows through a merge. Transp Res 37B:521–540

Kuwahara M (1990) Equilibrium queuing patterns at a two-tandem bottleneck during the morning peak. Transp Sci 24:217–229

Lighthill MJ, Whitham JB (1955) On kinematic waves: I. Flow modeling in long rivers, II. A theory of traffic flow on long crowded roads. In Proceedings of the Roy Society A 229:291–345

Lo HP, Szeto WY (2002) A cell-based variational inequality formulation of the dynamic user optimal assignment problem. Transp Res 36B:421–443

Mahmassani H, Herman R (1984) Dynamic user equilibrium departure time and route choice on idealized traffic arterials. Transp Sci 18:362–384

Nagurney A (1993) Network economics: A variational inequality approach. Kluwer, Norwell, MA, USA

Newell G (1987) The morning commute for non-identical travelers. Transp Sci 21:74–88

Nguyen S, Dupius C (1984) An efficient method for computing traffic equilibria in networks with asymmetric transportation costs. Transp Sci 18:185–202

Nie Y (2006) A variational inequality approach for inferring dynamic origin-destination travel demands. PhD dissertation, University of California, Davis. http://www.civil.northwestern.edu/people/nie-diss.pdf

Nie Y, Zhang HM (2005) Equilibrium analysis of the morning commute problem: numerical solution procedures. Journal of Mathematical and Computer Modeling, under review

Ran B, Hall RW, Boyce D (1996) A link-based variational inequality model for dynamic departure time/route choice. Transp Res 30B:31–46

Richards PI (1956) Shockwaves on the highway. Operation Research 4:42–51

Sheffi Y (1985) Urban transportation networks. Prentice-Hall, New Jersey

Small KA (1982) The scheduling of consumer activities: work trips. Am Econ Rev 72:467–479

Smith MJ (1993) A new dynamic traffic model and the existence and calculation of dynamic user equilibria on congested capacity-constrained road networks. Transp Res 26B:49–63

Szeto WY, Lo HP (2004) A cell-based simultaneous route and departure time choice model with elastic demand. Transp Res 38B:593–612

Tong CO, Wong SC (2000) A predictive dynamic traffic assignment model in congested capacity-constrained road networks. Transp Res 34B:625–644

Vickrey W (1969) Congestion theory and transport investment. Am Econ Rev 59:251–261

Wardrop JG (1952) Some theoretical aspects of road traffic research. In Proceedings of the Institute of Civil Engineers, Part II 1:325–378

Wei BW, Tobin RL, Friesz TL, Bernstein D (1995) A discrete time, nested cost operator approach to the dynamic network user equilibrium problem. Transp Sci 29:79–92

Wei BW, Tobin RL, Carey M (2002) The existence, uniqueness and computation of an arc-based dynamic network user equilibrium formulation. Transp Res 36B:897–918

Yang H, Huang H-J (1997) Analysis of the time-varying pricing of a bottleneck with elastic demand using optimal control theory. Transp Res 31B:425–440

Zhang HM, Nie Y (2005) Modeling network flow with and without link interaction: properties and implications. Paper presented at the 84th annual meeting of the Transportation Research Board

Zhu DL, P Marcotte (1993) Modified descent direction methods for solving the monotone variational inequality problem. Oper Res Lett 14:111–120