https://dx.doi.org/10.1016/j.ejor.2023.04.008

An Alternating Direction Method of Multipliers for Solving User Equilibrium Problem

Submitted by

Zhiyuan Liu, Ph.D., Professor
Jiangsu Key Laboratory of Urban ITS, Jiangsu Province Collaborative Innovation Center of Modern Urban Traffic Technologies, School of Transportation, Southeast University, China
E-mail: zhiyuanl@seu.edu.cn

Xinyuan Chen, Ph.D., Assistant Professor
College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, China
E-mail: xinyuan.chen@nuaa.edu.cn

Jintao Hu, Ph.D. Candidate
Department of Civil Engineering, The University of Hong Kong, Hong Kong
E-mail: u3006516@connect.hku.hk

Shuaian Wang, Ph.D., Professor
Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong, China
E-mail: wangshuaian@gmail.com

Kai Zhang, Ph.D. Candidate
Jiangsu Key Laboratory of Urban ITS, Jiangsu Province Collaborative Innovation Center of Modern Urban Traffic Technologies, School of Transportation, Southeast University, China
E-mail: zhangk2019@seu.edu.cn

AND

Honggang Zhang, Ph.D. Candidate
Jiangsu Key Laboratory of Urban ITS, Jiangsu Province Collaborative Innovation Center of Modern Urban Traffic Technologies, School of Transportation, Southeast University, China
E-mail: 230208829@seu.edu.cn

Corresponding Author:
Xinyuan Chen

February 2022

Abstract

This paper introduces a new parallel computing algorithm to address the user equilibrium (UE) problem. Searching for efficient solution algorithms for UE has been a recurring study subject in transportation research and has attracted much attention in past decades. Existing solution algorithms can be classified into three categories: link-based, path-based, and origin-based. This paper introduces an alternating direction method of multipliers (ADMM) algorithm that is different from these categories. Based on the origin-based formulation of UE problem, an equivalent problem is proposed which eliminates the flow conservation conditions through the augmented Lagrangian function. In order to make use of the ADMM, the network links should be grouped into different blocks, where the links in the same block are disconnected. This link grouping problem falls into the category of edge-coloring problem in graph theory, and it follows the Vizing theorem. A novel approach is developed for the link grouping problem. For links in the same block, we have a separable subproblem, which is solved in parallel by the gradient projection algorithm. Numerical experiments are conducted to validate the proposed algorithm, which shows its computation efficiency.

Keywords: Traffic Assignment, User Equilibrium, Parallel Computing, Alternating Direction Method of Multipliers, Edge-coloring Problem

1. Introduction

Traffic assignment is a fundamental tool to analyze the travel behavior of network users in the transportation network, and widely applied in the strategic/master planning of the urban transport systems. It forecasts network-wide traffic flow distribution patterns and the corresponding travel impedance based on given travel demand and infrastructure supply, which helps transport operators to better understand the role of transport infrastructure and develop sustainable transport systems. Although vast extensions have been developed, the user equilibrium condition proposed by Wardrop (1952) is still a cornerstone of traffic equilibrium problems. Applications based on user equilibrium (UE) traffic assignment problem in larger scale transport networks are largely limited because of the overwhelming computation burden (Bar-Gera, 2002; Nie, 2010). To further provide more timely and effective decision support, designing efficient solution algorithms for the UE problem has been a recurring research theme in transportation science and has attracted much attention in the past decades, which can be mainly classified as the link-based algorithms (Florian et al., 1987; LeBlanc et al., 1975; Mitradjieva and Lindberg, 2013), path-based algorithms (Jayakrishnan et al., 1994; Larsson and Patriksson, 1992; Xie et al., 2018), and origin-based solution algorithms (Bar-Gera, 1999, 2002; Nguyen, 1974; Nie, 2010, 2012; Xie and Nie, 2019; Xie et al., 2013). However, these algorithms are mainly designed for sequential computing.

In the past decades, benefitting from the rapid evolution of semiconductor technique, the performance of computers is growing exponentially. The Moore's law predicts that the number of transistors in a dense integrated circuit doubles about every two years. A major approach of achieving advances in computing processors is the parallel computing techniques. Currently, a wide range of high-performance laptops and desktops are very cost-efficient in the market. Therefore, utilizing these parallel resources to accelerate the convergence rate of UE problem becomes a timely and significant research direction.

The main challenge of applying parallel computing approach to solve the UE problem is that the mathematical formulation of UE cannot be directly partitioned into independent subproblems. To overcome this issue, this paper aims to utilize the methodology of ADMM (Boyd et al., 2011), a recently

prevailing and advanced method in the field of convex optimization, to effectively partition and solve the UE problem in parallel.

1.1. Literature review

The UE condition is originally proposed by Wardrop (1952), and known as the Wardrop's first principle. It describes the travelers' behavior in the transport network, i.e., all travelers with the same trip origin and destination (OD) have identical travel time, and no one can unilaterally reduce his/her travel time by changing routes. Beckmann et al. (1956) firstly transformed the UE condition into an equivalent nonlinear convex programming problem which enables a variety of efficient algorithms to solve this problem (Bar-Gera, 2002; Chen et al., 2013; Jayakrishnan et al., 1994; Larsson et al., 1993; Larsson and Patriksson, 1992; LeBlanc et al., 1975; Nie, 2010, 2012; Xie and Nie, 2019), to name a few.

The algorithms to address the Beckmann's transformation and its variations can be categorized into three lines. The first line of algorithms operates in the space of link flows (Chen et al., 2002; Fukushima, 1984a; LeBlanc et al., 1975). Frank-Wolfe algorithm is the most well-known and widely applied link-based solution algorithm, which is first introduced by LeBlanc et al. (1975). It is known for the simplicity of implementation and low requirement of computer memory. However, the algorithm has unsatisfactory performance in the vicinity of the optimum (Chen et al., 2020a). Several improvements of Frank-Wolfe algorithm have been made. Powell and Sheffi (1982) and Weintraub et al. (1985) modified the step size determinization method. Fukushima (1984a) modified the descent direction by using the information obtained from linearized subproblems in previous iterations. Florian et al. (1987) proposed an accelerated version by adapting the PARTAN direction. Mitradjieva and Lindberg (2013) improved Frank-Wolfe algorithm with conjugate and bi-conjugate direction. Besides the modification of descent direction and step size, Chen et al. (2002) improved the efficiency of Frank-Wolfe with the Gauss-Seidel iteration method.

The second line of algorithms operates in the space of path flows (Chen and Jayakrishnan, 1998; Jayakrishnan et al., 1994; Larsson and Patriksson, 1992; Xie et al., 2018), which has a unified algorithmic framework. Namely, they decompose the Beckmann's transformation into a series of OD-based subproblems, and then address them in the spirit of Gauss-Seidel iteration method. Typically, the column generation step is required to restrict the size of solution space, because the paths in the transport

network are numerous and it is impossible to enumerate and store each path in the computer. The major difference between various path-based algorithms is the way of solving OD-based subproblems. The first path-based algorithm was proposed by Dafermos and Sparrow (1969), which is known as path equilibration algorithm. The main idea is to move path flows from the longest path to the shortest path. Fukushima (1984b) formulated a Lagrangian dual problem with regards to the path-based UE formulation, and developed a subgradient-based algorithm to solve the dual traffic assignment problem. Their proposed Lagrangian dual problem proposed is nonsmooth convex and cannot be solved in parallel. Larsson and Patriksson (1992) solved an OD-based auxiliary problem, and updated the solutions through a convex combination of extreme points. Jayakrishnan et al. (1994) proposed the gradient projection algorithm to solve the OD-based subproblems. Chen and Jayakrishnan (1998) further studied the effect of two path flow update policies: Jacobi and Gauss-Seidel iteration methods. Florian et al. (2009) proposed a similar projected gradient method to address the decomposed subproblems, which moves path flows from longer paths to shorter ones. Another similar method called improved social pressure algorithm was proposed by Kumar and Peeta (2011). Xie et al. (2018) proposed a greedy path-based algorithm in which the OD-based subproblem is approximated by a quadratic programming problem, and then addressed by a greedy approach. Another break-through achieved by Xie et al. (2018) is the adoption of an intelligent scheme to filter and skip the OD pairs that already have sufficient accuracy. This idea was later extended by Babazadeh et al. (2020) and Galligari and Sciandrone (2019).

The third line of algorithms is the origin-based algorithms which can be further classified into three types. The first type is the network simplex (NS) algorithm, which is the first origin-based algorithm to address UE problem (Nguyen, 1974). Zheng and Peeta (2014) extended the NS by allowing the spanning tree to contain links that could either from or to the origin. Bar-Gera (2010) proposed a similar origin-based algorithm, which is termed paired alternative segments (PAS). A PAS was established as two completely disjoint path segments with the same starting and ending nodes. Xie and Xie (2016) improved this algorithm with a new PAS identification method and associated each PAS with only one origin to speed up the convergence efficiency. The second type of origin-based algorithm was proposed by Bar-Gera (2002), which sequentially solves a series of node-based subproblems defined on the “bush”

rooted at each origin by a gradient projection method. Nie (2010) studied and compared a class of bush-based algorithms. Nie (2012) presented some corrections to Bar-Gera's algorithm (Bar-Gera, 2002). Gentile (2014) proposed an alternative greedy algorithm to replace the gradient projection to address the node-based subproblems, which is named local user cost equilibrium algorithm. Xie and Nie (2019) improved the bush-based algorithm with entropy maximization subproblems. The path flow solution satisfies the proportionality condition in general networks. The third type of origin-based algorithm originates from Dial's algorithm B (Dial, 2006). Algorithm B simultaneously finds the minimal and maximal path trees for a given origin and then equalizes the bush rooted at the origin by moving flows from the longest path to the shortest ones. Zheng and Peeta (2014) proposed an intelligent rule, i.e., cost scaling based successive approximation, for algorithm B.

The studies of parallel computing on UE problem are relatively sparse. Feijoo and Meyer (1988) proposed a piecewise-linear approximation method to address the non-separable convex optimization problem and applied it on a multi-commodity traffic equilibrium problem. The approximated objective function is separable which permits parallel computation. Numerical examples indicate an increased efficiency in parallel computing approach with an increase in problem size. Later, Chen and Meyer (1988) made an extension by adding a trust region approach. Larsson and Patriksson (1992) proposed a disaggregate simplicial decomposition (DSD) algorithm in which the decomposed subproblems can be solved in parallel. Patriksson (1994) summarized a general parallel algorithm framework, partial linearization, of the abovementioned research. Recently, Jafari et al. (2017) decomposed the transport network into smaller sub-networks and executed the user equilibrium algorithm on each sub-network parallelly. A common idea of these studies is to construct a separable auxiliary problem and address it in parallel. However, additional operations, e.g., line search, are required to keep the accuracy of descent tendency during the iteration. New methods are needed to generate independent subproblem while avoiding the additional expensive line search operation.

1.2. Objectives and contributions

Recently, the alternating direction method of multipliers, i.e., ADMM, algorithm has received much attention in the field of convex optimization for parallel computing of large-scale continuous optimization, as a combination of the method of multipliers and block coordinate descent method (Boyd

et al., 2011). This study introduces a new algorithmic framework to address the UE problem following the spirit of ADMM.

We first propose an equivalent reformulation of the origin-based formulation of UE problem, in which the node-based flow conservation constraints are eliminated through the augmented Lagrangian function. The new formulation only has nonnegative constraints, which can contribute to some efficient solution methods.

Then, in the spirit of ADMM, the reformulated model is decomposed into a series of block-based subproblems. In other words, grouping the network links into several blocks, fixing other blocks' origin-based link flows, and updating only one block's origin-based link flows at a time. We find that if the links in the same block are disconnected with each other, the block-based subproblems are separable with respect to (w.r.t.) links, and thus can be addressed in parallel. We note that the parallelization is achieved through alternatively solving the block-based subproblems, rather than constructing approximation subproblems.

The decomposed problem is a typical convex nonlinear optimization problem with nonnegative constraints, which enables a class of efficient algorithms to address it. In this study, we propose a tailored gradient projection algorithm to address the separated link-based subproblems. In addition, to ensure the network links in the same block are distinct with each other, we proceed to utilize the characteristic of transport network and propose a novel method to group the network links.

In summary, the contributions of this study are twofold. Firstly, based on the origin-based formulation of UE problem, a new ADMM-based algorithmic framework is proposed which decomposes this equivalent formulation into a series of block-based subproblems. By grouping the disconnected links into the same block, the block-based subproblems become a summation of independent link-based subproblems, which can be addressed in parallel. The independent link-based subproblems are generated by alternatively fixing the origin-based link flows, rather than constructing approximation problems. Second, for the subproblems of the ADMM-based method, a novel method is first proposed for the grouping of the links, which is in nature an edge-coloring problem; then for the links in each block that are independent, an efficient gradient projection method is adopted to solve the optimization problems with only non-negative constraints.

The remainder of this paper is organized as follows: in the next section, we present the mathematical formulations of UE condition and introduce the method of multipliers. In Section 3, the alternating direction method of multipliers to address the UE problem is introduced. Section 4 introduces a network partition method to group network links into several blocks. Section 5 discusses the method to address the decomposed link-based subproblems. Numerical experiments are conducted in Section 6. Finally, Section 7 concludes this study.

2. Model formulation

The UE condition is usually formulated as a convex program which has two typical formulations according to the definition of flow conservation conditions. For the sake of presentation, both formulations are introduced in this section. A full list of the notation is provided in the Appendix A.

2.1. A convex programming model

Consider a strongly connected transport network $G=(N, A)$, where N denotes the set of nodes and A denotes the set of links. Let W denote the set of OD pairs and q^{od} the travel demand for $(o,d)\in W$. Let K^{od} denote the set of paths between OD pair (o,d) and f_k^{od} the flow on path $k\in K^{od}$. v_a denotes the flow on link $a\in A$. $\delta_{a,k}^{od}$ denotes the link-path incidence relationship, which equals 1 when path $k\in K^{od}$ uses link a, and 0, otherwise. The travel time function of link $a\in A$ is denoted as $t_a(v_a)$, which is strictly increasing and continuously differentiable.

A widely used mathematical formulation of UE condition is formulated as convex programming problem:

[MP-UE]

$$\min \mathrm{Z}_1=\sum_{a\in A}\int_0^{v_a} t_a(w)\,dw \tag{1}$$

subject to

$$\sum_{k\in K^{od}} f_k^{od}=q^{od},\ \forall od\in W \tag{2}$$

$$f_k^{od}\geq 0,\ \forall k\in K^{od}, od\in W \tag{3}$$

$$v_a = \sum_{od \in W} \sum_{k \in K^{od}} f_k^{od} \delta_{a,k}^{od}, \forall a \in A. \quad (4)$$

The Karush-Kuhn-Tucker (KKT) conditions (Sheffi, 1985) show its equivalence to the UE condition which is written as:

$$f_k^{od}\left(c_k^{od} - \varphi^{od}\right) = 0, \quad \forall k \in K^{od}, od \in W \quad (5)$$

$$f_k^{od} \geq 0, \left(c_k^{od} - \varphi^{od}\right) \geq 0, \quad \forall k \in K^{od}, od \in W \quad (6)$$

where φ^{od} are the Lagrange multipliers associated with the flow conservation conditions in Eqs. (2), which represents the shortest travel time between OD pair (o,d); c_k^{od} denotes the travel cost of the path k between OD pair (o,d). It is clear to see that the model (1)-(4) is strictly convex in terms of the link flow variables $v_a, a \in A$, which can be efficiently solved by a link/path-based algorithm, e.g., the Frank-Wolfe and the Gradient Projection method.

2.2. An origin-based model

It can be seen that the above [MP-UE] model is not separable w.r.t. link or path flow variables, mainly due to the flow conservation constraints (2) and (4), considering that the objective function (1) is separable w.r.t link flows. These constraints, in theory, could be converted into the objective function via the Lagrangian duality theory. However, since the constraints (2) and (4) are built on paths (rather than links or nodes), and the number of paths is tremendously large in an urban transport network, it is difficult to be directly handled by Lagrangian duality or ADMM.

To deal with this issue, an origin-based model can be adopted, which is originally proposed by Beckmann (Beckmann et al., 1956). Wherein, the flow conservation constraints are defined on nodes rather than paths. Let O denote the set of origins, i.e., $O \subseteq N$, and W^o denote the set of OD pairs originating from $o \in O$. $i(a)$ is the tail node of link $a \in A$. $h(a)$ stands for the head node of link $a \in A$. v_a^o is denoted as the traffic flow on link $a \in A$ originating from $o \in O$. $\mathbf{v}$ is a vector of v_a^o which is defined as $\mathbf{v} := \left(v_a^o, \forall o \in O, a \in A\right)$. Then, the flow on link a equals $v_a = \sum_{o \in O} v_a^o$. The origin-based model is then given as:

[OB-UE]

$$\min Z_2 = \sum_{a\in A} \int_0^{\sum_{o\in O} v_a^o} t_a(w)\,dw \tag{7}$$

subject to

$$\sum_{\substack{a\in A,\\ i(a)=n}} v_a^o - \sum_{\substack{a\in A,\\ h(a)=n}} v_a^o = g_n^o, \forall o\in O, n\in N \tag{8}$$

$$v_a^o \geq 0, \forall o\in O, a\in A \tag{9}$$

where constraints (8) define the node-based flow conservation equations and g_n^o is defined as:

$$g_n^o = \begin{cases} \sum_{od\in W^o} q^{od}, n = o \\ -q^{od}, n = d \\ 0, \text{otherwise} \end{cases}, \quad \forall o\in O, n\in N. \tag{10}$$

The idea of solving this origin-based model [OB-UE] is to decompose it into a series of restricted origin based subproblems and solve them sequentially by the Gauss-Seidel iteration scheme (Nie, 2010). In this study, we aim to use this origin-based model and employ its Lagrangian relaxation, in order to cope with the independent flow conservation equations (2) and (4).

For the sake of presentation, we give a vector form of the node-based conservation equation (8):

$$C\cdot \mathbf{v} - \mathbf{g} = 0 \tag{11}$$

where $\mathbf{v}$ is a vector of origin-based link flows, $\mathbf{v} := \left(v_a^o, \forall a\in A, o\in O\right)$, C is the coefficient matrix w.r.t. $\mathbf{v}$, and $\mathbf{g}$ is a vector of origin-based travel demand, $\mathbf{g} := \left(g_n^o, \forall o\in O, n\in N\right)$. Let $H_n^o(v) := \sum_{\substack{a\in A,\\ i(a)=n}} v_a^o - \sum_{\substack{a\in A,\\ h(a)=n}} v_a^o - g_n^o, \forall o\in O, n\in N$. The Lagrangian function for [OB-UE] is:

$$L_0(\mathbf{v},\boldsymbol{\lambda}) = \sum_{a\in A} \int_0^{\sum_{o\in O} v_a^o} t_a(w)\,dw + \sum_{o\in O}\sum_{n\in N} \lambda_n^o H_n^o(v) \tag{12}$$

where $\lambda_n^o \in \boldsymbol{\lambda}$ denotes the Lagrange multiplier or dual variable associated with the equality constraint with specified origin o and node n in Eqs. (8), and $\boldsymbol{\lambda} := \left(\lambda_n^o, \forall o\in O, n\in N\right)$ denotes its vector form. Then, the dual problem of [OB-UE] is:

$$\max_{\boldsymbol{\lambda}} \inf_{\mathbf{v}\geq 0} L_0(\mathbf{v},\boldsymbol{\lambda}) \tag{13}$$

It can be easily proven that the objective function (7) is convex but not strict convex w.r.t. the origin-based link flows $\mathbf{v}$. Considering that the constraints (9) are affine functions, the strong duality holds (Boyd and Vandenberghe, 2004). Thus, the optimal value of the primal equals the optimal value

of the dual problem. We can obtain a primal optimal point $\mathbf{v}^*$ from a dual optimal point $\boldsymbol{\lambda}^*$ (Boyd et al., 2011):

$$\min_{\mathbf{v}\geq 0} L_0(\mathbf{v},\boldsymbol{\lambda}^*). \tag{14}$$

2.3. The augmented Lagrangian model

The dual problem can be commonly solved by some gradient-based algorithms, e.g., the dual ascent method. However, these algorithms usually require strict convexity of the primal model (Boyd et al., 2011), which does not hold for (7). Thus, the augmented Lagrangian is further adopted for [OB-UE] to guarantee the convergence of solution algorithms:

$$L_\rho(\mathbf{v},\boldsymbol{\lambda}) = \sum_{a\in A}\int_0^{\sum_{o\in O} v_a^o} t_a(w)dw + \sum_{o\in O}\sum_{n\in N}\lambda_n^o H_n^o(v) + \frac{\rho}{2}\sum_{o\in O}\sum_{n\in N}\left(H_n^o(v)\right)^2 \tag{15}$$

where $\rho > 0$ denotes the penalty parameter.

Then, a new model [ROB-UE] can be provided for the UE problem:

[ROB-UE]

$$\max_{\boldsymbol{\lambda}} \inf_{\mathbf{v}\geq 0} L_\rho(\mathbf{v},\boldsymbol{\lambda}). \tag{16}$$

Note that the augmented Lagrangian shown in Eq. (17) is strictly convex w.r.t. $\mathbf{v}$, which is the objective function of the [ROB-UE] model. Proof of the convexity is provided in Appendix B.

In theory, we could use *method of multipliers* (Boyd et al., 2011) to solve the [ROB-UE], which consists of iterating the updates:

$$\mathbf{v}^{(i+1)} := \arg\min_{\mathbf{v}\geq 0} L_\rho\left(\mathbf{v},\boldsymbol{\lambda}^{(i)}\right) \tag{18}$$

$$\boldsymbol{\lambda}^{(i+1)} := \boldsymbol{\lambda}^{(i)} + \rho\left(C\cdot\mathbf{v}^{(i+1)} - \mathbf{g}\right) \tag{19}$$

where ρ is taken as the step size when updating $\boldsymbol{\lambda}$ and i denotes the number of iterations.

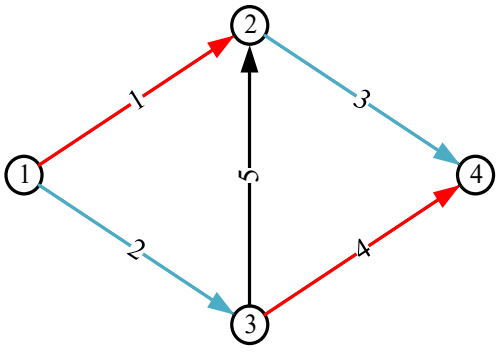


Figure 1 Illustrative network.

However, the update step of primal variables in the *method of multipliers,* i.e. step (18), still cannot be solved in parallel, because the node-based flow conservation equations $H_n^o(v) := \sum_{\substack{a\in A,\\ i(a)=n}} v_a^o - \sum_{\substack{a\in A,\\ h(a)=n}} v_a^o - g_n^o$ are not independent for different *o* and different *n* in the network. We use the toy-size example shown in Figure 1 for illustration. We can see that in this example, at node 2, the links 1, 3 and 5 are combined to fulfill one node-based flow conservation equations. Thus, these three links are interdependent. Likewise, since the entire transport network is strongly connected, all the links are connected with other ones, and thus we cannot partition the network into separable parts. That is to say, the node-based flow conservation has caused the interdependence of origin-based link flows $\mathbf{v}$, which are the decision variables of our addressed problems.

To further cope with this new problem (independence of origin-based link flows $\mathbf{v}$), we proceed to use the ADMM, which is elaborated in the next section.

3. The alternating direction method of multipliers for UE

The basic idea of ADMM is to split the variables $\mathbf{v}$ into different parts. For illustration, we use mutually exclusive two parts, termed $\mathbf{v}_{B_1}$ and $\mathbf{v}_{B_2}$. Thus, the node-based flow conservation equation/constraint (11) can be reformulated as:

$$C_1 \cdot \mathbf{v}_{B_1} + C_2 \cdot \mathbf{v}_{B_2} - \mathbf{g} = 0 \tag{20}$$

where C_1 and C_2 are the coefficient matrix w.r.t. $\mathbf{v}_{B_1}$ and $\mathbf{v}_{B_2}$. Then, the variables can be updated in the following order:

$$\mathbf{v}_{B_1}^{(i+1)} := \arg\min_{\mathbf{v}_{B_1}\geq 0} L_\rho^{B_1}\left(\mathbf{v}_{B_1}, \mathbf{v}_{B_2}^{(i)}, \boldsymbol{\lambda}^{(i)}\right) \tag{21}$$

$$\mathbf{v}_{B_2}^{(i+1)} := \arg\min_{\mathbf{v}_{B2}\geq 0} L_\rho^{B_2}\left(\mathbf{v}_{B_1}^{(i+1)}, \mathbf{v}_{B_2}, \boldsymbol{\lambda}^{(i)}\right) \tag{22}$$

$$\boldsymbol{\lambda}^{(i+1)} := \boldsymbol{\lambda}^{(i)} + \rho\left(C_1 \cdot \mathbf{v}_{B_1}^{(i+1)} + C_2 \cdot \mathbf{v}_{B_2}^{(i+1)} - \mathbf{g}\right) \tag{23}$$

where the subscript B_1 in $L_\rho^{B_1}$ indicates that only $\mathbf{v}_{B_1}$ is decision variable, while all the other flows (i.e., $\mathbf{v}_{B_2}$) are fixed. All the other terms in $L_\rho^{B_1}$ are the same with Eq. (15).

Thereby, the links in the network should be grouped into different blocks, such that links in each block are independent. For a particular block, the augmented Lagrangian function (15) can then be solved/updated in parallel.

Assume the link set A is partitioned into P blocks, where links in each block are independent. Let B_p denote the set of links in p th block, where $1 \le p \le P$. Then, the origin-based link flow variable $\mathbf{v}$ can also be split into P blocks, i.e., $\mathbf{v}_{B_1}, \mathbf{L}, \mathbf{v}_{B_P}$. For block B_p, the block-based augmented Lagrangian function can be obtained by fixing other variables besides $\mathbf{v}_{B_p}$:

$$L_\rho^{B_p}\left(\mathbf{v}_{B_1}^{(i+1)},...,\mathbf{v}_{B_{p-1}}^{(i+1)},\mathbf{v}_{B_p},\mathbf{v}_{B_{p+1}}^{(i)},...,\mathbf{v}_{B_P}^{(i)},\boldsymbol{\lambda}^{(i)}\right). \tag{24}$$

As the links in the same block do not connect with each other, the block-based subproblem (24) can be further expressed as the summation of independent link-based subproblems:

$$L_\rho^{B_p}\left(\mathbf{v}_{B_1}^{(i+1)},...,\mathbf{v}_{B_{p-1}}^{(i+1)},\mathbf{v}_{B_p},\mathbf{v}_{B_{p+1}}^{(i)},...,\mathbf{v}_{B_P}^{(i)},\boldsymbol{\lambda}^{(i)}\right)=\sum_{a\in B_p} L_\rho^{B_p,a}\left(\mathbf{v}_a\right) \tag{25}$$

where $\mathbf{v}_a$ is a vector $\mathbf{v}_a := \left(v_a^o, \forall o \in O\right)$. Namely, the block-based subproblem can be decomposed into a series of independent link-based subproblems which is suitable for parallel computing.

3.1. The block-based subproblem

The link blocking plan is a key issue in using ADMM. Herein, a successful link blocking plan can group the links with no overlapped tail/head nodes into the same block/set. For a particular link blocking plan, it gives rise to a particular block-based subproblem when using the ADMM. In this sub-section, we use the network example in Figure 1 to illustrate these two concepts, link blocking plan and block-based subproblem. Then, a standard formulation is provided for the block-based subproblem of ADMM in this sub-section 3.1 in order to complete the whole picture of ADMM. The solution algorithm for link blocking is discussed in Section 4.

We can see that link 1 has overlapped nodes with link 2, 3 and 5, therefore, none of link 2, 3 nor link 5 can be grouped into the same block with link 1. Following this blocking plan, link 1 and link 4 are grouped into one block/set, say, B_1; links 2 and 3 are grouped into another block/set B_2; link 5 connects all the other four links, thus it is separately grouped into B_3. Accordingly, the links in each

block are not connected with each other. Let $\mathbf{v}_{B_1}$, $\mathbf{v}_{B_2}$ and $\mathbf{v}_{B_3}$ denote the vector of origin-based link flows in block B_1, B_2 and B_3, respectively.

Taking the subproblem for $\mathbf{v}_{B_1}$ as example, with fixed Lagrange multipliers $\boldsymbol{\lambda}$ and fixed primal variables $\mathbf{v}_{B_2}$ and $\mathbf{v}_{B_3}$, the block-based subproblem can be given as follows:

$$\begin{aligned}\min_{\mathbf{v}_{B_1}\geq 0} L_\rho^{B_1}\left(\mathbf{v}_{B_1},\mathbf{v}'_{B_2},\mathbf{v}'_{B_3},\boldsymbol{\lambda}'\right)=&\sum_{a\in B_1}\int_0^{\sum_{o\in O}v_a^o} t_a(w)dw+\sum_{a\in B_2}\int_0^{\sum_{o\in O}(v')_a^o} t_a(w)dw+\sum_{a\in B_3}\int_0^{\sum_{o\in O}(v')_a^o} t_a(w)dw\\&+\sum_{o\in O}\sum_{n\in N}\lambda_n^o H_n^o(v)+\frac{\rho}{2}\sum_{o\in O}\sum_{n\in N}\left(H_n^o(v)\right)^2\end{aligned} \tag{26}$$

where $\mathbf{v}'_{B_2}$ and $\mathbf{v}'_{B_3}$ denote fixed origin-based link flows and $\boldsymbol{\lambda}'$ denotes the fixed feasible Lagrange multipliers. Therefore, the block-based subproblem (26) contains a series of constant terms, i.e., the second and third terms. The fourth and fifth terms are combinations of variables and constants. Recall that $H_n^o(v)$ in subproblem (26) denotes the inflow-outflow pattern of node n w.r.t. origin o, which can be written as:

$$H_n^o(v):=\left(\sum_{\substack{a\in B_1,\\ i(a)=n}} v_a^o+\sum_{\substack{a\in B_2,\\ i(a)=n}}(v')_a^o+\sum_{\substack{a\in B_3,\\ i(a)=n}}(v')_a^o\right)-\left(\sum_{\substack{a\in B_1,\\ h(a)=n}} v_a^o+\sum_{\substack{a\in B_2,\\ h(a)=n}}(v')_a^o+\sum_{\substack{a\in B_3,\\ h(a)=n}}(v')_a^o\right)-g_n^o. \tag{27}$$

Based on a successful link blocking rule, for the $H_n^o(v),\forall o\in O,n\in N$ in Eq. (26), there should exist at most one link's origin-based variables $\mathbf{v}_a:=\left(v_a^o,\forall o\in O\right),a\in B_1$. For example, assuming there exists only one OD pair (1, 4) in the illustrative network, the subscript o in $H_n^o(v)$ becomes 1. We can list the $H_n^o(v)$ of all the four nodes in Figure 1 for illustration as follows:

$$H_1^1(v)=\left(v_1^1+\sum_{\substack{a\in B_2,\\ i(a)=1}}(v')_a^1+\sum_{\substack{a\in B_3,\\ i(a)=1}}(v')_a^1\right)-\left(\sum_{\substack{a\in B_2,\\ h(a)=1}}(v')_a^1+\sum_{\substack{a\in B_3,\\ h(a)=1}}(v')_a^1\right)-g_1^1 \tag{28}$$

$$H_2^1(v)=\left(\sum_{\substack{a\in B_2,\\ i(a)=2}}(v')_a^1+\sum_{\substack{a\in B_3,\\ i(a)=2}}(v')_a^1\right)-\left(v_1^1+\sum_{\substack{a\in B_2,\\ h(a)=2}}(v')_a^1+\sum_{\substack{a\in B_3,\\ h(a)=2}}(v')_a^1\right)-g_2^1 \tag{29}$$

$$H_3^1(v)=\left(v_4^1+\sum_{\substack{a\in B_2,\\ i(a)=3}}(v')_a^1+\sum_{\substack{a\in B_3,\\ i(a)=3}}(v')_a^1\right)-\left(\sum_{\substack{a\in B_2,\\ h(a)=3}}(v')_a^1+\sum_{\substack{a\in B_3,\\ h(a)=3}}(v')_a^1\right)-g_3^1 \tag{30}$$

$$H_4^1(v)=\left(\sum_{\substack{a\in B_2,\\ i(a)=4}}(v')_a^1+\sum_{\substack{a\in B_3,\\ i(a)=4}}(v')_a^1\right)-\left(v_4^1+\sum_{\substack{a\in B_2,\\ h(a)=4}}(v')_a^1+\sum_{\substack{a\in B_3,\\ h(a)=4}}(v')_a^1\right)-g_4^1. \tag{31}$$

Clearly, Eqs. (28)-(31) demonstrate that each $H_n^o(v)$ in the subproblem (26) has at most one link's origin-based variable. That is, v_1^1 and v_4^1 for $H_1^1(v)$ and $H_3^1(v)$, respectively, and none variable exists in $H_2^1(v)$ and $H_4^1(v)$.

For the ease of presentation, let z denote the constant part of the integral terms in Eq. (26) and e_n^o denote the constant part in H_n^o. Then, in the illustrative case, i.e., Figure 1, we have:

$$z=\int_0^{(v')_2^1}t_2(w)dw+\int_0^{(v')_3^1}t_3(w)dw+\int_0^{(v')_5^1}t_5(w)dw$$

$$e_1^1=\left(\sum_{\substack{a\in B_2,\\ i(a)=1}}(v')_a^1+\sum_{\substack{a\in B_3,\\ i(a)=1}}(v')_a^1\right)-\left(\sum_{\substack{a\in B_2,\\ h(a)=1}}(v')_a^1+\sum_{\substack{a\in B_3,\\ h(a)=1}}(v')_a^1\right)-g_1^1$$

$$e_2^1=\left(\sum_{\substack{a\in B_2,\\ i(a)=2}}(v')_a^1+\sum_{\substack{a\in B_3,\\ i(a)=2}}(v')_a^1\right)-\left(\sum_{\substack{a\in B_2,\\ h(a)=2}}(v')_a^1+\sum_{\substack{a\in B_3,\\ h(a)=2}}(v')_a^1\right)-g_2^1$$

$$e_3^1=\left(\sum_{\substack{a\in B_2,\\ i(a)=3}}(v')_a^1+\sum_{\substack{a\in B_3,\\ i(a)=3}}(v')_a^1\right)-\left(\sum_{\substack{a\in B_2,\\ h(a)=3}}(v')_a^1+\sum_{\substack{a\in B_3,\\ h(a)=3}}(v')_a^1\right)-g_3^1$$

$$e_4^1=\left(\sum_{\substack{a\in B_2,\\ i(a)=3}}(v')_a^1+\sum_{\substack{a\in B_3,\\ i(a)=3}}(v')_a^1\right)-\left(\sum_{\substack{a\in B_2,\\ h(a)=3}}(v')_a^1+\sum_{\substack{a\in B_3,\\ h(a)=3}}(v')_a^1\right)-g_4^1.$$

Thus,

$$\begin{aligned}&\min_{\mathbf{v}_{B_1}\ge 0}L_\rho^{B_1}\left(\mathbf{v}_{B_1},\mathbf{v}'_{B_2},\mathbf{v}'_{B_3},\boldsymbol{\lambda}'\right)=\int_0^{v_1^1}t_1(w)dw+\int_0^{v_4^1}t_4(w)dw+z\\&+\left(\lambda_1^1\left(v_1^1+e_1^1\right)+\lambda_2^1\left(e_2^1-v_1^1\right)+\lambda_3^1\left(v_4^1+e_3^1\right)+\lambda_4^1\left(e_4^1-v_4^1\right)\right)\\&+\frac{\rho}{2}\left(\left(v_1^1+e_1^1\right)^2+\left(e_2^1-v_1^1\right)^2+\left(v_4^1+e_3^1\right)^2+\left(e_4^1-v_4^1\right)^2\right)\end{aligned} \tag{32}$$

where only v_1^1 and v_4^1 are decision variables. Because links 1 and 4 in B_1 have no overlap nodes, the block-based subproblem (32) is separable w.r.t. links 1 and 4. We proceed to illustrative this idea. Let $L_\rho^{B_1,a}$ denote the separated link-based augmented Lagrangian function and specify the link $a=1$, we have:

$$L_\rho^{B_1,1}\left(v_1^1\right)=\int_0^{v_1^1} t_1(w)dw+\lambda_1^1\left(v_1^1+e_1^1\right)+\lambda_2^1\left(e_2^1-v_1^1\right)+\frac{\rho}{2}\left(v_1^1+e_1^1\right)^2+\frac{\rho}{2}\left(e_2^1-v_1^1\right)^2. \tag{33}$$

Similarly, specifying $a=4$, we have:

$$L_\rho^{B_1,4}\left(v_4^1\right)=\int_0^{v_4^1} t_4(w)dw+\lambda_3^1\left(v_4^1+e_3^1\right)+\lambda_4^1\left(e_4^1-v_4^1\right)+\frac{\rho}{2}\left(v_4^1+e_3^1\right)^2+\frac{\rho}{2}\left(e_4^1-v_4^1\right)^2. \tag{34}$$

Then, subproblem (32) can be written as:

$$\min_{\mathbf{v}_{B_1}\geq 0} L_\rho^{B_1}\left(\mathbf{v}_{B_1},\mathbf{v}'_{B_2},\mathbf{v}'_{B_3},\boldsymbol{\lambda}\right)=\min_{\mathbf{v}_{B_1}\geq 0} L_\rho^{B_1,1}\left(v_1^1\right)+L_\rho^{B_1,4}\left(v_4^1\right). \tag{35}$$

That means the subproblem (32) is the summation of link-based subproblems (33) and (34) which can be solved in parallel.

The ADMM works when we alternatively fix the link flow variables in $\mathbf{v}_{B_1},\mathbf{v}_{B_2},\mathbf{v}_{B_3}$ and solve the block-based subproblems subsequently, followed by a dual update of Lagrange multiplier $\boldsymbol{\lambda}$. In other words,

$$\mathbf{v}_{B_1}^{(i+1)}=\arg\min_{\mathbf{v}_{B_1}\geq 0} L_\rho^{B_1}\left(\mathbf{v}_{B_1},\mathbf{v}_{B_2}^{(i)},\mathbf{v}_{B_3}^{(i)},\boldsymbol{\lambda}^{(i)}\right) \tag{36}$$

$$\mathbf{v}_{B_2}^{(i+1)}=\arg\min_{\mathbf{v}_{B_2}\geq 0} L_\rho^{B_2}\left(\mathbf{v}_{B_1}^{(i+1)},\mathbf{v}_{B_2},\mathbf{v}_{B_3}^{(i)},\boldsymbol{\lambda}^{(i)}\right) \tag{37}$$

$$\mathbf{v}_{B_3}^{(i+1)}=\arg\min_{\mathbf{v}_{B_3}\geq 0} L_\rho^{B_3}\left(\mathbf{v}_{B_1}^{(i+1)},\mathbf{v}_{B_2}^{(i+1)},\mathbf{v}_{B_3},\boldsymbol{\lambda}^{(i)}\right) \tag{38}$$

$$\boldsymbol{\lambda}^{(i+1)}=\boldsymbol{\lambda}^{(i)}+\rho\left(C\cdot\mathbf{v}^{(i+1)}-\mathbf{g}\right). \tag{39}$$

3.2. General procedures of ADMM in solving UE

In this sub-section, we proceed to provide the general procedures of ADMM method for solving [ROB-UE]. Similar to the procedures of solving the illustrative example in Section 3.1, the ADMM solves the block-based subproblems (24) as follows:

$$\mathbf{v}_{B_1}^{(i+1)}:=\arg\min_{\mathbf{v}_{B_1}\geq 0} L_\rho^{B_1}\left(\mathbf{v}_{B_1},\mathbf{v}_{B_2}^{(i)},\mathbb{L},\mathbf{v}_{B_P}^{(i)},\boldsymbol{\lambda}^{(i)}\right) \tag{40}$$

……

$$\mathbf{v}_{B_p}^{(i+1)} := \arg\min_{\mathbf{v}_{B_p}\geq 0} L_\rho^{B_p}\left(\mathbf{v}_{B_1}^{(i+1)},\mathrm{L}\ ,\mathbf{v}_{B_{p-1}}^{(i+1)},\mathbf{v}_{B_p},\mathbf{v}_{B_{p+1}}^{(i)},\mathrm{L}\ ,\mathbf{v}_{B_P}^{(i)},\boldsymbol{\lambda}^{(i)}\right) \tag{41}$$

……

$$\mathbf{v}_{B_P}^{(i+1)} := \arg\min_{\mathbf{v}_{B_P}\geq 0} L_\rho^{B_P}\left(\mathbf{v}_{B_1}^{(i+1)},\mathrm{L}\ ,\mathbf{v}_{B_{P-1}}^{(i+1)},\mathbf{v}_{B_P},\boldsymbol{\lambda}^{(i)}\right) \tag{42}$$

$$\boldsymbol{\lambda}^{(i+1)} = \boldsymbol{\lambda}^{(i)} + \rho\left(C\cdot\mathbf{v}^{(i+1)} - \mathbf{g}\right). \tag{43}$$

As aforementioned, the parallelism of [ROB-UE] is achieved in each block by fixing the origin-based link flows of other blocks. Each iteration updates a block of the origin-based link flows in parallel, while keeping other variables as constants. Note that ADMM is a variation of the *method of multipliers*, where a single Gauss-Seidel iteration method is used to pass over $\mathbf{v}_{B_p}, p=1,\mathrm{L}\ ,P$, rather than the usual joint minimization (Boyd et al., 2011). The overall procedure of ADMM is shown in the following:

Algorithm 1. ADMM for UE

1 Initialize:
2 Iterate:
3 Update primal variables sequentially according to Eqs. (40)–(42):
4 Update $\mathbf{v}_{B_1}^{(i+1)}$ according to Eq. (40)
5 ……
6 Update $\mathbf{v}_{B_p}^{(i+1)}$ according to Eq. (41)
7 ……
8 Update $\mathbf{v}_{B_p}^{(i+1)}$ according to Eq. (42)
9 Update dual variable $\boldsymbol{\lambda}$ according to Eq. (43)
10 Go back to Step 3 until the termination criterion is satisfied.

For the termination criterion, note that due to the flow conservation conditions are not strictly obeyed in ADMM, the commonly used termination criterion in UE, e.g., relative gap, is no longer suitable in this algorithm. Alternative termination measurements could be the primal/dual residual defined in ADMM or the following absolute value of relative gap ($|RG|$):

$$\left|RG\right| = \left|1 - \frac{\sum_{od \in W} \varphi^{od} q^{od}}{\sum_{a \in A} v_a t_a\left(v_a\right)}\right| \tag{44}$$

where φ^{od} is the minimal travel time/cost between OD pair (o,d), i.e., travel cost of the shortest path between OD pair (o,d). When only the single OD pair is considered, φ^{od} is the optimal value, and it can be calculated by using any widely adopted shortest path algorithm, e.g., the label-correcting algorithm.

4. Solution algorithms for link blocking

In this section, we proceed to discuss the link blocking problem, which is in nature a network partition or edge coloring problem in graph theory (Bollobás, 2013). The network links need to be separated into several blocks such that the links in the same block do not connect with each other. Thereby, the links in each block are independent and their subproblems (45) can be solved in parallel.

In order to enhance the level of parallelism of the [ROB-UE] model, the number of links in each block is expected to be larger, i.e., the total number of blocks should be smaller. Such a requirement tally with the objective of edge coloring problem. The definition of edge coloring problem is to color the links of a network such that no two incident edges have the same color. The objective of edge coloring problem is to find/use the minimum number of colors required for each network.

The type of transport networks defined in Section 2.1 falls into the category of directed multigraph graph in graph theory (Bollobás, 2013). A simple graph implies that any two nodes in the network are connected by at most one link; while in a multigraph, two nodes may be connected by more than one link. Three basic definitions of the edge-coloring problem are presented as follows:

Definition 1 (Degree): The degree (or valency) of a vertex/node of a graph is the number of edges that are incident to the vertex, denoted by $\deg(n), \forall n \in N$. The maximum degree of a graph G, denoted by $\Delta(G)$, is the largest degree of the vertices in the graph.

Definition 2 (Chromatic index): Given a graph, the minimum required number of colors for the edges is named the chromatic index of the graph, denoted by $\chi(G)$.

Definition 3 (Multiplicity): The maximum number of edges in any bundle of parallel edges of a graph is called the multiplicity, denoted by $u(G)$. For a transport network, due to the exist of bi-directional road links, usually $u(G)=2$.

It is clear that the minimum number of blocks in UE problem equals the Chromatic index of the network. In edge coloring problem, a key issue is to get the Chromatic index of any given network, for which the most well-known theory is the Vizing theorem (Bollobás, 2013). Vizing theorem states that for simple graphs, the graph G may be edge colored using either $\Delta(G)$ colors or $\Delta(G)+1$ colors, i.e., $\Delta(G)\le\chi(G)\le\Delta(G)+1$.

However, transportation networks are not simple graphs but commonly multigraphs. Shannon (1949) shows that the chromatic index of a multigraph G should follow $\chi(G)\le(3/2)\Delta(G)$. In addition, it is further revealed that for any multigraph, $\chi(G)\le\Delta(G)+u(G)$ (Bollobás, 2013). For example, for a common transport network with the multiplicity $u(G)=2$ and its intersections have no more than 6 arms, i.e. $\Delta(G)=12$, we can infer that the upper bound and lower bound of its chromatic index: $12\le\chi(G)\le14$.

In summary, for both simple graphs and multigraphs, the theoretical upper bound of their chromatic index are provided. However, for a particular transport network, to find its exact chromatic index is an NP-hard problem (Holyer, 1981). Namely, unless P=NP, it is impossible to determine in polynomial time whether the chromatic index of a transport network is 12, 13 or 14. Thus, in this section, we propose an efficient heuristic for the link blocking problem of any transportation network which takes the greedy rule. The algorithm is summarized as follows:

Algorithm 2. A greedy-based algorithm for link blocking

---

1 Initialize a node set $S=\varnothing$ to store checked nodes and a node set $S'=\varnothing$ to store

queuing nodes. Let $c(a)$ denote the color (number) of link a and set $c(a)=0$ for each link $a \in A$. Initialize a set $B(n)=\varnothing$ to stored color of links connected with node n.

2 Select and push an initial node into S'.

3 While $S' \neq \varnothing$ do:

4 Select the first node n in S' as the checking node, and delete it from S'.

5 For each link a connected with n do:

6 If the tail node $t(a)$ or the head node $h(a)$ of a is not in S and not equals n:

7 Push it into S'.

Color the links connect with checking n: Line 8-16

8 For each link a connected with n do:

9 If $c(a)>0$:

10 Push $c(a)$ into set $B(t(a))$ and $B(h(a))$.

11 Else:

12 $c=1$

13 While $c(a)$ in $B(t(a))$ or $B(h(a))$

14 $c(a)=c(a)+1$

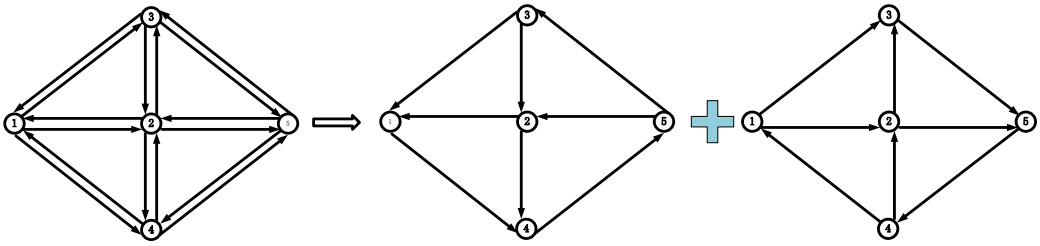
15 $c(a)=c$

16 Push c into $B(t(a))$ and $B(h(a))$.

17 Push node n in S.

In the above algorithm, $c(a)$ is the assigned number (color) of link a, where $c(a)=0$ means link a is uncolored, and each positive integer represents a distinct color. $B(n)$ denotes the set w.r.t. node n to store the assigned colors of adjacent links. The proposed method is able to efficiently group the network links into a relatively small number of blocks.

Note that although the performance of this heuristic is acceptable, it cannot guarantee a solution with at most $\Delta(G)+u(G)$ blocks. To address this problem, we can propose another algorithm by utilizing the special feature of transport network. Here, we use an illustrative example to clarify this idea. As shown in Figure 2(a), the transport network is a typical bi-directional 2-multigraph. We observe that Figure 2(a) is a combination of two simple sub-networks, Figure 2(b) and Figure 2(c).



(a) Transport network **(b) Simple network 1** **(c) Simple network 2**

Figure 2 An illustrative example of coloring a transport network.

Considering the algorithm for coloring the edges of simple network is well developed (Bollobás, 2013), we separate the transport network to two simple sub-networks. Then, we can easily color simple networks by utilizing a known algorithm, e.g, Misra & Gries's algorithm (Misra and Gries, 1992). Note that this algorithm is the fastest known almost-optimal algorithm for edge coloring. Figure 2 (a) can be colored by combining the solutions together. In a simple network, the number of colors is at most $\lceil \Delta(G)/2 \rceil + 1$ (note that $\Delta(G)$ is the maximum degree of original transport network G, $\lceil \Delta(G)/2 \rceil$ is the maximum degree of a simple sub-network). Thus, the number of colors for G is at most $\Delta(G)+u(G)$, which is a very tight bound. Herein, $u(G)$ denotes the multiplicity of transport network G; usually $u(G)=2$. The formal description of the almost-optimal algorithm is described as follows:

Algorithm 3. An almost-optimal algorithm for link-blocking

1	Reduce the transport network into two separate simple networks
2	Use an existing algorithm (e.g., Misra & Gries's algorithm) to color the simple networks
3	Combine the blocks of simple networks together with distinct colors

5. Solution method for the link-based subproblem

5.1. The link-based subproblem

In this section, we proceed to solve the link-based minimization problem, which is the most important/time-consuming subproblem of ADMM:

$$\min_{\mathbf{v}_a \geq 0, a \in B_p} L_\rho^{B_p,a}\left(\mathbf{v}_a\right) = \int_0^{\sum_{o\in O} v_a^o} t_a\left(w\right)dw + \sum_{o\in O}\left[\lambda_{i(a)}^o\left(H_{i(a)}^o\left(v\right)\right) + \lambda_{h(a)}^o\left(H_{h(a)}^o\left(v\right)\right) + \frac{\rho}{2}\left(H_{i(a)}^o\left(v\right)\right)^2 + \frac{\rho}{2}\left(H_{h(a)}^o\left(v\right)\right)^2\right]. \quad (45)$$

Note that when solving the above problem for link $a \in B_p$ at the $i+1$ th iteration, the origin-based link flows of $\mathbf{v}_{B_1}^{(i+1)},...,\mathbf{v}_{B_{p-1}}^{(i+1)}$ are fixed as the newly updated values in $i+1$ th iteration, while those of $\mathbf{v}_{B_{p+1}}^{(i)},...,\mathbf{v}_{B_P}^{(i)}$ and $\boldsymbol{\lambda}^{(i)}$ are fixed as the old values in the previous iteration i. The other links in B_p are independent with link $a \in B_p$, thus these subproblems can be solved in parallel by the same algorithm. Hence, this section only takes the subproblem for link $a \in B_p$ for illustration.

Note that the feasible region of model (45) is defined on the non-negative orthant, thus the projection operation on the feasible region becomes very straightforward (Bertsekas, 1974). Namely, for any given origin-based link flow $\mathbf{v}_a = \left(v_a^o, \forall o \in O\right)$, we have:

$$P_+\left[\mathbf{v}_a\right] = \left(\max\left(0, v_a^o\right), \forall o \in O\right) \quad (46)$$

where $P_+\left[\cdot\right]$ is the projection operation. This fact inspires us to use an efficient projection type method for solving inequality constrained minimization problem (45), which is elaborated in the following Sub-section 5.1.

5.2. A tailored gradient projection method

The gradient projection algorithm is widely used in solving traffic assignment problems. Jayakrishnan et al. (1994) proposed a path-based UE solution algorithm, by adopting Goldstein-Levitin-Poljak gradient projection algorithm formulated by Bertsekas (1974). This method is further extended by Chen and Jayakrishnan (1998) via taking the Gauss-Seidel scheme to replace the Jacobi scheme and achieve a much higher convergence performance. Following the work of Chen and Jayakrishnan (1998), a tailored gradient projection algorithm is developed in this sub-section for the link-based subproblem.

The flows are updated towards the descent direction of the second-order approximation of a transformed objective function. If the solution point is out of the feasible region, a projection is made to restrict the solution to the feasible region boundaries. The basic update equation is written as follows:

$$v_a^{o(i+1)} = \max\left[0, v_a^{o(i)} - \alpha\left(s_a^{o(i)}\right)^{-1}\left(d_a^{o(i)}\right)\right], \forall o \in O, a \in A \tag{47}$$

where i is the iteration counter, α is the step size, $s_a^{o(i)}$ is the positive definite scalar (the second-order derivative), and $d_a^{o(i)}$ is the first order derivatives which represent the descent direction. Bertsekas et al. (1984) suggested using α=1 for all iteration i, which alleviates the difficulty of finding a proper range of the step size and it has been found to be a very robust choice in earlier research on different network sizes and topologies.

Then, we proceed to derive the first-order derivative d_a^o and the second-order derivative s_a^o of the link-based subproblem (45). Recall that H_n^o denotes the node-based flow conservation function, i.e., $H_n^o(v) := \sum_{\substack{a\in A,\\ i(a)=n}} v_a^o - \sum_{\substack{a\in A,\\ h(a)=n}} v_a^o - g_n^o$, and e_n^o denotes the constant part in $H_n^o(v)$. Then, the first order condition of (45) w.r.t. v_a^o is expressed as:

$$d_a^o = \partial L_\rho^{B_p,a}(\mathbf{v}_a)/\partial v_a^o = t_a\left(\sum_{o\in O} v_a^o\right) + 2\rho v_a^o + \left(\rho e_{i(a)}^o - \rho e_{h(a)}^o + \lambda_{i(a)}^o - \lambda_{h(a)}^o\right). \tag{48}$$

The second-order derivative is expressed as:

$$s_a^o = \partial^2 L_\rho^{B_p,a}(\mathbf{v}_a)/\partial\left(v_a^o\right)^2 = t_a'\left(\sum_{o\in O} v_a^o\right) + 2\rho. \tag{49}$$

Based on the above discussion, the complete algorithmic steps are summarized as follows:

Algorithm 4. A tailored gradient projection algorithm for link-based subproblem

1 Set $j = 0$. Assign zero flow to origin-based link flows $v_a^{o(j)}, \forall a \in A, o \in O$.

2 For each origin o do:

3 Update travel time t_a of link a, $t_a^{(j)} = t_a\left(\sum_{o\in O} v_a^{o(j)}\right)$.

4 Update second order derivative of the objective function, $s_a^{o(j)} = t_a'\left(\sum_{o\in O} v_a^{o(j)}\right) + 2\rho$.

5 Update the first order derivative of the objective function, $d_a^{o(j)} = t_a\left(\sum_{o\in O} v_a^{o(j)}\right) + 2\rho v_a^o + \rho e_{i(a)}^o$

$-\rho e^{o}_{h(a)}+\lambda^{o}_{i(a)}-\lambda^{o}_{h(a)}$.

6 Update origin-based link flow, $v_a^{o(j+1)}=\max\left[0, v_a^{o(j)}-\alpha\cdot\left(s_a^{o(j)}\right)^{-1}\cdot d_a^{o(j)}\right]$.

7 Update link flow, $v_a^{(j+1)}=v_a^{(j)}+v_a^{o(j+1)}-v_a^{o(j)}$.

8 Let $j=j+1$. Go back to Step 2 until the termination criterion is satisfied.

As to the termination criterion, two criteria can be used to measure the accuracy of the solution. One is the absolute gradient gap (AG), which is written as:

$$AG^{(j)}=\sum_{o\in O}\left|d_a^{o(j)}\cdot v_a^{o(j)}\right|. \tag{50}$$

Another is the error of AG between the j th and $j-1$ th iteration (EAG), which is written as:

$$EAG^{(j)}=\left|AG^{(j)}-AG^{(j-1)}\right|. \tag{51}$$

6. Numerical examples

In this section, the proposed ADMM algorithm are tested on three network examples to show its convergence performance and compare with alternative algorithms.

All the experiments are conducted on a Windows 10 64-bit computer with AMD 4800H 2.9 GHz CPU and 16G RAM. The algorithms are coded by the authors in C++ using Microsoft Visual Studio Code. Each experiment is executed multiple times until the CPU time becomes stable. We use the absolute value of relative gap, $|RG|$ in Eq. (44), as the precision measurement to reflect the convergence performance of these algorithms. The BPR function $t_a=t_a^0\left(1+\alpha\left(v_a/C_a\right)^{\beta}\right)$ is used as the link impedance function, where t_a^o is the free flow travel time and C_a is the capacity on link a. The parameters α and β are set as 0.15 and 4, respectively.

6.1. A small network

We first use the small network in Figure 1 to verify the proposed method. The link attributes are provided in Table 1. The OD travel demand between node 1 and 4 is assumed to be 60. The penalty ρ in ADMM is set as 1, without loss of generality. The convergence trend is plotted in Figure 3. Table 2 reports the convergence process of link flows and their equilibrium flow pattern.

Table 1 Link attributes of the illustrative network.

Link ID	Tail	Head	Free flow travel time	Capacity	Block ID
1	1	2	3	10	1
2	1	3	2	10	2
3	2	4	4	10	2
4	3	2	1	10	3
5	3	4	5	10	1

Table 2 Evaluation of link flows

No. of iterations	Link 1	Link 2	Link 3	Link 4	Link 5
0	0	0	0	0	0
1	32.7178	**34.9552**	**31.2048**	1.5124	29.4172
2	30.6535	33.8593	31.3418	2.2790	29.9892
3	29.8462	33.1369	31.3836	2.4292	29.8156
4	29.3573	32.6004	31.3279	2.4394	29.6922
5	29.0557	32.2356	31.2492	2.4215	29.5860
…					
88	28.4808	31.5191	30.8365	2.3556	29.1634

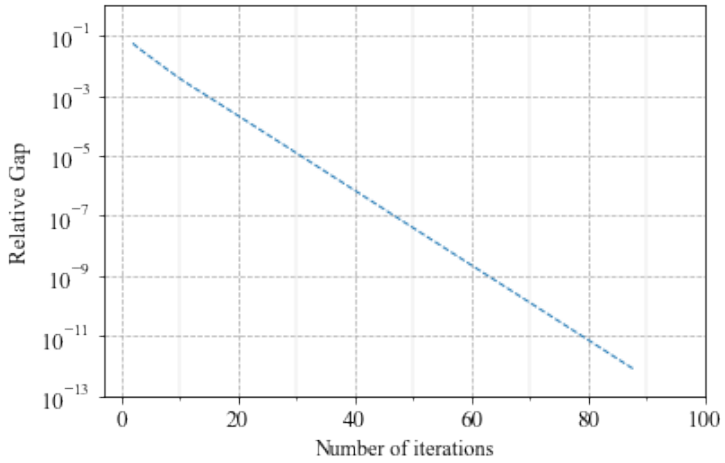


Figure 3 Convergence trend of ADMM in the illustrative network

The origin-based link flows are updated by the gradient projection algorithm (Algorithm 4 in Section 5.2). Here, we take the Link 3 in the 1st iteration of ADMM as an example to illustrate the computation process of solving link-based subproblem (45). To update v_3, the descent direction of origin-based link flow v_3^1 is calculated. As per Eq. (48) and (49), the first and second order derivative of Link 3 is calculated:

$$d_3^1 = 60.8907+31.2048\times 2+\left(-32.1660-30.0107-3.0254-59.3778\right) = -1.2795 \tag{52}$$

$$s_3^1 = 7.2925+2\times 1 = 9.2925. \tag{53}$$

where the superscript 1 of d_3^1 implies the OD pair 1 and the subscript 3 of d_3^1 represents the Link 3. Then, the origin-based link flow v_3^1 can be updated:

$$v_3^1 = \max\left[0, v_3^1 - d_3^1/s_3^1\right] = 31.3425. \tag{54}$$

A projection operation is operated in Eq. (54). The origin-based link flow would be revised to zero if it becomes negative. When the AG (see Eq. (50)) is less than 10^{-9}, the algorithm terminates. The whole computation process of solving link-based subproblem (45) is summarized in Table 3, which shows the gradient projection algorithm can rapidly achieve the predetermined precision within a few iterations.

Table 3 Computation process of link-based subproblem on Link 3

No. of iterations	1st derivative	2nd derivative	AG	Link flow
0	-	-	-	31.2048
1	–1.2795	9.2925	-	31.3425
2	0.0066	9.3895	0.2089	31.3418
3	1.7823×10^{-7}	9.3890	0	-

Meanwhile, Link 2 in the same block is solved simultaneously. The computation process is summarized in Table 4, which also demonstrates the efficiency of gradient projection algorithm.

Table 4 Computation process of link-based subproblem on Link 2

No. of iterations	1st derivative	2nd derivative	AG	Link flow
0	-	-	-	34.9552
1	7.5498	7.1252	-	33.8956
2	0.2419	6.6731	8.1930	33.8594
3	0.0002	6.6582	0.0092	33.8593
4	3.436×10^{-10}	6.6581	1.163×10^{-8}	33.8593
5	1.421×10^{-14}	6.6581	4.811×10^{-13}	-

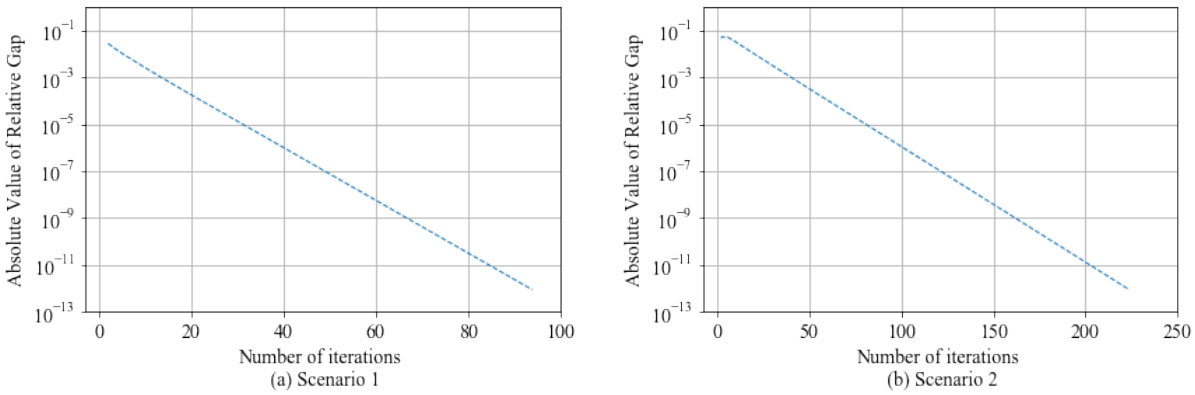


Figure 4 Convergence process of ADMM in the illustrative network with various scenarios

We proceed to test the case of multiple OD pairs. The OD travel demand between node 1 and 2 is assumed to be 10, which is added to the trip demand table (Scenario 1). The results also show the convergence trend, as shown in Figure 4(a). In addition, we proceed to expand the trip demand table by adding the 10-unit OD demand between 3 to 4 (Scenario 2). The convergence trend is also verified, as shown in Figure 4(b).

6.2. Sioux-Falls network

We proceed to test the proposed methods using the most widely adopted Sioux-Falls network, as shown in Figure 5. Data of the network is accessible at the website: https://github.com/bstabler/TransportationNetworks. Using the proposed algorithm provided in Section 4, the resultant link blocking plan is provided in Table 5. As maximum degree of Sioux-Falls network is $\Delta = 10$ (node 10 is a five-way intersection with 10 connecting links), the Chromatic index of Sioux Falls network is between 10 and 12. Table 5 shows that the network links are grouped into 10 blocks. It implies that the links have been grouped into minimal number of blocks, which is clearly an optimal solution. We then test the performance of proposed ADMM method. Firstly, for the accuracy of the ADMM method, we compare its resultant link flows with the optimal solution (indicated as Best-Known Solution), which is shown in Table C1 at Appendix C. It clearly shows that the proposed ADMM method has achieved the optimal UE solution.

Secondly, as shown in Figure 6, we further compare the newly proposed ADMM-based method with some other existing/well-adopted solution methods, e.g., Frank-Wolfe (FW) and Bar-Gera's origin-based algorithm (OBA). The flow conservation condition is strictly obeyed in the FW and OBA; therefore, the relative gap is always positive. However, ADMM solution does not strictly hold this

condition/constraint in early iterations. Thus, the absolute value of relative gap is used to reflect the solution's precision. For the parameters in the ADMM algorithm, we set the $\rho = 0.008$, $AG = 10^{-5}$, and $EAG = 10^{-3}$ in this section.

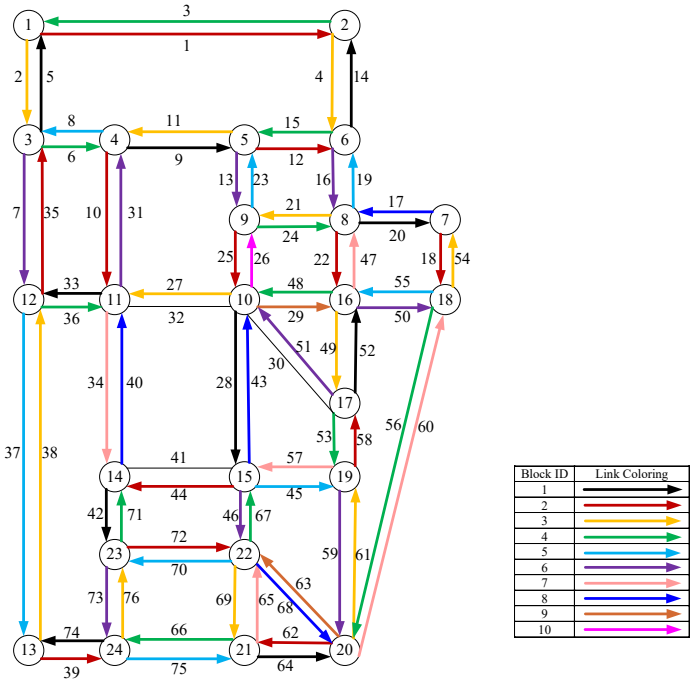


Figure 5 Layout of Sioux-Falls network

Table 5 Link blocking pattern

Block ID	Index of Member Links
1	5,9,14,20,28,33,42,52,64,74
2	1,10,12,18,22,25,35,39,44,58,62,72
3	2,4,11,21,27,38,41,49,54,61,69,76
4	3,6,15,24,36,48,53,56,66,67,71
5	8,19,23,32,37,45,55,70,75
6	7,13,16,31,46,50,51,59,73
7	30,34,47,57,60,65
8	17,40,43,68
9	29,63
10	26

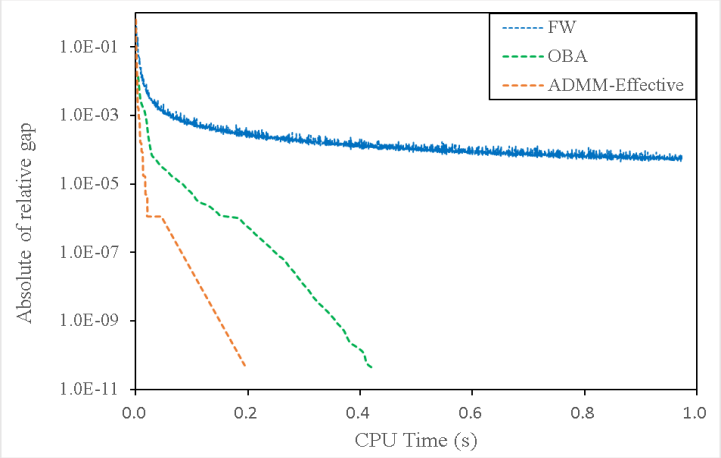


Figure 6 Convergence performance comparison on Sioux-Falls network.

The concept of effective computing time (Chen et al., 2020b) is used to reflect the performance upper bound of ADMM. Specifically, suppose we have a parallel computing block c which has l independent computing elements. Then, in an ideal parallel computing environment with sufficient processors, the effective computing time t_c^{eff} equals the elements' maximum computing time. Namely,

$$t_c^{eff} = \max\left(t_q^c, \forall q \in (1,\ldots,l)\right) \tag{55}$$

where q represents an independent computing element in task c. Effective computing time t_c^{eff} provides a theoretical measurement of the of parallel computing. Additionally, in line with Eq. (44), it can be found that the shortest path between OD pair is required when using the relative gap, which is not necessary in the framework of the ADMM algorithm. Hence, the cost of calculating the relative gap is not included in the effective computing time. In Figure 6, the yellow dotted line represents the convergence performance of the ADMM algorithm in the ideal parallel computing environment. As it shows that the performance of ADMM is superior to the FW algorithm. Both ADMM and OBA rapidly achieve the $|RG| \le 10^{-10}$ within 1 second and ADMM is a bit faster than OBA. The results indicate that ADMM is an efficient algorithm to address UE traffic assignment problem.

We further analyze the influence of number of blocks on the computation efficiency of the ADMM algorithm, and the numerical experiment on Sioux-Falls network is shown in Figure 7. Herein, the ADMM_10_Block, ADMM_11_Block and ADMM_12_Block satisfy the Vizing theory that a solution with at most $\Delta(G)+u(G)$ blocks. The ADMM_10_Block has a minimal number of blocks. In addition, we consider the particular case ADMM_13_Block where the number of grouped blocks

exceed $\Delta(G)+u(G)$. From the above numerical experiment, we find the ADMM_10_Block case has the best performance. However, ADMM_13_Block has the worst performance. It can be found the better performance is the case that fewer blocks are used in the ADMM algorithm.

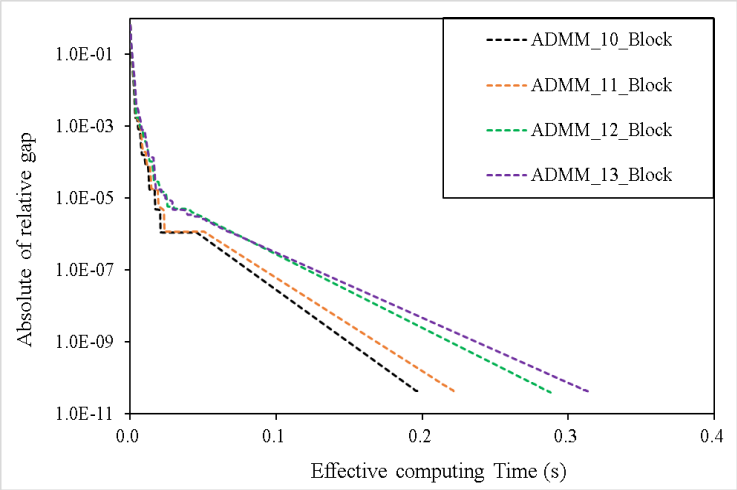


Figure 7 Influence of the number of blocks on Sioux-Falls network

6.3. Anaheim network

To further enhance the numerical analysis and verify the ADMM method, we employed a moderate Anaheim network (see Figure 8) to validate the proposed algorithm. This network comprises 416 nodes, 914 links and 1,406 OD pairs. Subsequently, the network links are grouped into 12 blocks. The ADMM algorithm is tested on Anaheim network to solve the user equilibrium traffic assignment problem. The computational results for different algorithms are shown in Figure 9. It can be found that the OBA algorithm takes about 3.6 seconds to achieve |RG|=1.0e-10. In contrast, the ADMM algorithm can converge to the same precision level much faster, which is accelerated by 48%. This moderate network example thus further underpins the effectiveness of proposed ADMM method.

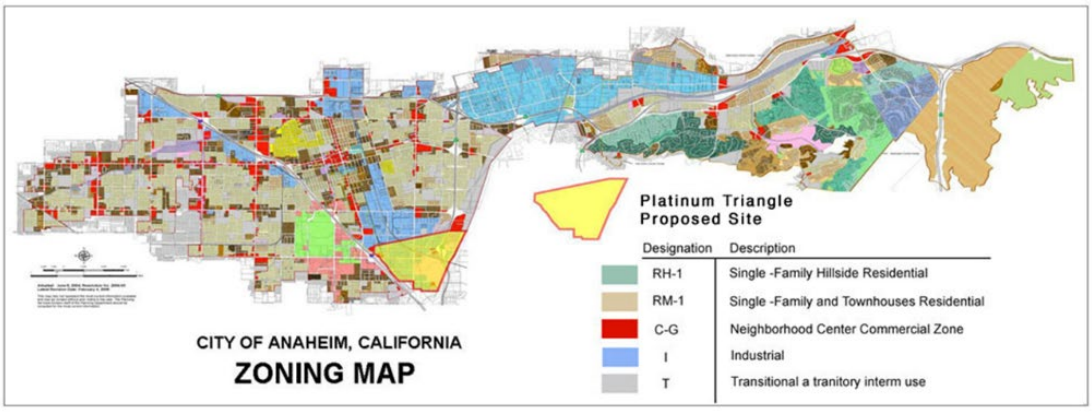


Figure 8 Layout of Anaheim network

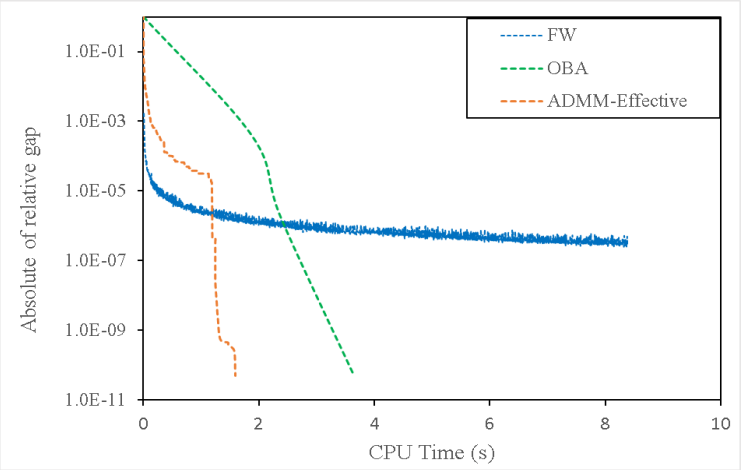


Figure 9 Algorithm performance comparison on Anaheim network

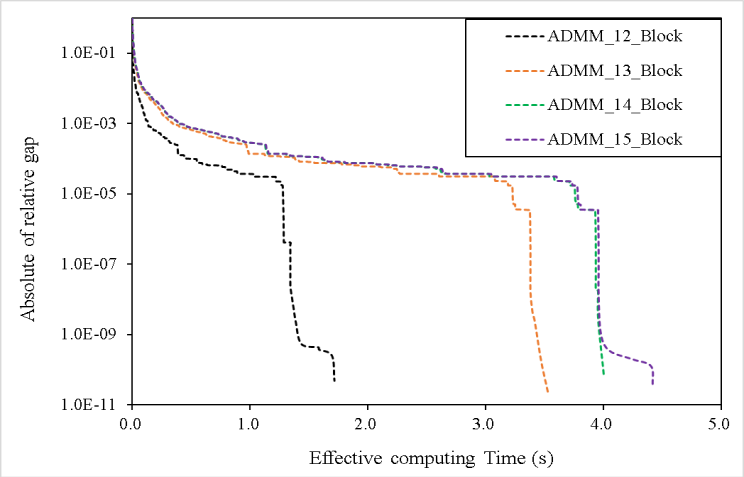


Figure 10 Influence of the number of blocks on Anaheim network

Likewise, we discuss the influence of the number of blocks on the convergence performance of the ADMM algorithm on Anaheim network, and the computation result is shown in Figure 10. As shown in Figure 10, it can be found that the ADMM_12_Block has the best performance. In contrast, the particular case ADMM_15_Block has the worst performance, in which the number of grouped blocks exceed $\Delta(G)+u(G)$. Based on the above analysis, we can conclude that the ADMM algorithm has better performance when fewer blocks are used.

6.4. Chicago-Sketch network

Thereafter, we employed a big size network (the Chicago Sketch network) to validate the proposed method, which comprises 933 nodes, 2950 links and 93513 OD pairs. Subsequently, based on the link blocking method proposed in this paper (see Section 4 for more details) , the network links are grouped

into 20 blocks. The ADMM algorithm is applied to solve the UE-TAP on Chicago-Sketch network. Therefore, for links in the same block, we have separable link-based subproblems which can be parallelly solved by the gradient projection algorithm (i.e., Algorithm 4). The computational results for different algorithms are shown as follows:

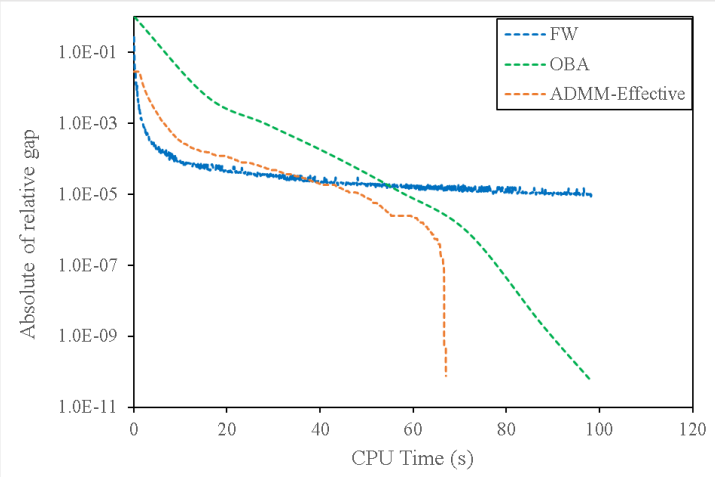


Figure 11 Algorithm performance comparison on the Chicago-Sketch network

As shown in Figure 11, the ADMM algorithm converges to the same precision level much faster compared with the OBA, which is accelerated by 51%. Therefore, the numerical experiment on a big size network further verifies the effectiveness of the proposed algorithm.

7. Conclusion

This study proposed an alternating direction method of multipliers to address the UE problem. A new model decomposition approach was proposed based on the origin-based formulation. The flow conservation conditions were eliminated through the augmented Lagrangian function. Based on the spirit of alternative direction, we grouped the disconnected links into a number of blocks and updated the origin-based link flow block by block. The proposed method lends itself well for parallel computing. An almost-optimal algorithm for link blocking was proposed to group the network links into different blocks. Compared with existing advanced solution algorithms, the proposed algorithm is suitable for paralleling computing. Because a series of independent link-based subproblems can be generated by fixing the link flows on connected links. Following the spirit of block-coordinate, the newly updated link flow and cost information can be timely used to update links flow in the next block. Therefore, the

convergence rate is also enhanced. Three numerical experiments were conducted to validate the proposed method. The results show that the performance of ADMM is superior to some existing algorithms, including FW and OBA. This study presented an initial step on the aspect of using ADMM for parallel computing of UE. More efforts will be explored to further accelerate the ADMM and also extend its use to other types of traffic assignment related problems.

Acknowledgement

This study is supported by the Key Project (No. 52131203) of National Natural Science Foundation of China.

References

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

Bar-Gera, H., 1999. Origin-based algorithms for transportation network modeling, *Civil Engineering*. University of Illinois at Chicago, Chicago, IL.

Bar-Gera, H., 2002. Origin-based algorithm for the traffic assignment problem. *Transportation Science* 36(4), 398–417.

Bar-Gera, H., 2010. Traffic assignment by paired alternative segments. *Transportation Research Part B: Methodological* 44(8), 1022–1046.

Beckmann, M., McGuire, C.B., Winsten, C.B., 1956. Studies in the economics of transportation, Yale University Press, Connecticut.

Bertsekas, D., Gafni, E., Gallager, R., 1984. Second derivative algorithms for minimum delay distributed routing in networks. *IEEE Transactions on Communications* 32(8), 911–919.

Bertsekas, D.P., 1974. On the Goldstein-Levitin-Polyak gradient projection method. *IEEE Transactions on Automatic Control* 21(2), 47–52.

Bollobás, B., 2013. *Modern graph theory*. Springer Science & Business Media, New York.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning* 3(1), 1–122.

Boyd, S., Vandenberghe, L., 2004. *Convex Optimization*. Cambridge university press, Cambridge, U.K.

Chen, A., Jayakrishnan, R., 1998. A path-based gradient projection algorithm: effects of equilibration with a restricted path set under two flow update policies, Institute of Transportation Studies, University of California, Irvine.

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

Chen, A., Xu, X., Ryu, S., Zhou, Z., 2013. A self-adaptive Armijo stepsize strategy with application to traffic assignment models and algorithms. *Transportmetrica A* 9(8), 695-712.

Chen, R.J., Meyer, R.R., 1988. Parallel optimization for traffic assignment. *Mathematical Programming* 42(1), 327-345.

Chen, X., Liu, Z., Kim, I., 2020a. A parallel computing framework for solving user equilibrium problem on computer clusters. *Transportmetrica A: Transport Science* 16(3), 550-573.

Chen, X., Liu, Z., Zhang, K., Wang, Z., 2020b. A parallel computing approach to solve traffic assignment using path-based gradient projection algorithm. *Transportation Research Part C: Emerging Technologies* 120, 102809.

Dafermos, S.C., Sparrow, F.T., 1969. The traffic assignment problem for a general network. *Journal of Research of the National Bureau of Standards B* 73(2), 91–118.

Dial, R.B., 2006. A path-based user-equilibrium traffic assignment algorithm that obviates path storage and enumeration. *Transp. Res. Part B: Methodological* 40(10), 917-936.

Feijoo, B., Meyer, R., 1988. Piecewise-linear approximation methods for nonseparable convex optimization. *Management Science* 34(3), 411–419.

Florian, M., Constantin, I., Florian, D., 2009. A New Look at Projected Gradient Method for Equilibrium Assignment. *Transportation Research Record* 2090(1), 10-16.

Florian, M., Guálat, J., Spiess, H., 1987. An efficient implementation of the "Partan" variant of the linear approximation method for the network equilibrium problem. *Networks* 17(3), 319–339.

Fukushima, M., 1984a. A modified Frank-Wolfe algorithm for solving the traffic assignment problem. *Transportation Research Part B: Methodological* 18(2), 169–177.

Fukushima, M., 1984b. On the dual approach to the traffic assignment problem. *Transp. Res. Part B: Methodological* 18(3), 235-245.

Galligari, A., Sciandrone, M., 2019. A computational study of path-based methods for optimal traffic assignment with both inelastic and elastic demand. *Comput. Oper. Res.* 103, 158-166.

Gentile, G., 2014. Local User Cost Equilibrium: a bush-based algorithm for traffic assignment. *Transportmetrica A: Transport Science* 10(1), 15–54.

Holyer, I., 1981. The NP-completeness of edge-coloring. *SIAM Journal on Computing* 10(4), 718–720.

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

Jayakrishnan, R., Wei, T.T., Prashker, J.N., Rajadhyaksha, S., 1994. A faster path-based algorithm for traffic assignment. *Transportation Research Record: Journal of the Transportation Research Board* 1443(1994), 75–83.

Kumar, A., Peeta, S., 2011. An improved social pressure algorithm for static deterministic user equilibrium traffic assignment problem, *Proceedings of 90th Annual Meeting of the Transportation Research Board*, Washington, DC.

Larsson, T., Migdalas, A., Partriksson, M., 1993. A partial linearization method for the traffic assignment problem. *Optimization* 28(1), 47–61.

Larsson, T., Patriksson, M., 1992. Simplicial decomposition with disaggregated representation for the traffic assignment problem. *Transportation Science* 26(1), 4–17.

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.

Misra, J., Gries, D., 1992. A constructive proof of Vizing's theorem. *Information Processing Letters* 41(3), 131-133.

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

Nguyen, S., 1974. An algorithm for the traffic assignment problem. *Transportation Science* 8(3), 203–216.

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

Nie, Y., 2012. A note on Bar-Gera's algorithm for the origin-based traffic assignment Problem. *Transportation Science* 46(1), 27–38.

Patriksson, M., 1994. *The Traffic Assignment Problem: Models and Methods*. Courier Dover Publications, VSP, Utrecht, The Netherlands.

Powell, W.B., Sheffi, Y., 1982. The convergence of equilibrium algorithms with predetermined step sizes. *Transportation Science* 16(1), 45–55.

Shannon, C.E., 1949. A theorem on coloring the lines of a network. *Journal of Mathematics and Physics* 28(1–4), 148–152.

Sheffi, Y., 1985. *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Prentice Hall, Englewood Cliffs, NJ.

Wardrop, J.G., 1952. Some theoretical aspects of road traffic research, *Proceedings of the institution of Civil Engineering, Part II*, London, pp. 325–378.

Weintraub, A., Ortiz, C., González, J., 1985. Accelerating convergence of the Frank-Wolfe algorithm. *Transportation Research Part B: Methodological* 19(2), 113–122.

Xie, J., Nie, Y., 2019. A new algorithm for achieving proportionality in user equilibrium traffic assignment. *Transportation Science* 53(2), 566–584.

Xie, J., Nie, Y., Liu, X., 2018. A greedy path-based algorithm for traffic assignment. *Transportation Research Record: Journal of the Transportation Research Board* 2672(48), 36–44.

Xie, J., Nie, Y., Yang, X., 2013. Quadratic approximation and convergence of some bush-based algorithms for the traffic assignment problem. *Transportation Research Part B: Methodological* 56, 15–30.

Xie, J., Xie, C., 2016. New insights and improvements of using paired alternative segments for traffic assignment. *Transportation Research Part B: Methodological* 93, 406–424.

Xie, X.Z., Ono, T., Nakano, S., Hirata, T., 2004. An improved algorithm for the nearly equitable edge-coloring problem. *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences* E87A(5), 1029-1033.

Zheng, H., Peeta, S., 2014. Cost scaling based successive approximation algorithm for the traffic assignment problem. *Transp. Res. Part B: Methodological* 68, 17-30.