An International Journal of Optimization
and Control: Theories & Applications
Vol.2, No.2, pp.173-186 (2012) © IJOCTA
ISSN:2146-0957 eISSN:2146-5703
http://www.ijocta.com

Finding Most Vital Links over Time in a Flow Network

Shahram MOROWATI-SHALILVAND and Javad MEHRI-TEKMEH

Faculty of Mathematical Science, University of Tabriz, Tabriz, Iran
Email: ShMorowati@live.com
Email: JMehri@tabrizu.ac.ir

(Received November 19, 2012; in final form June 12, 2012)

Abstract. This paper deals with finding most vital links of a network which carries flows over time (also called *"dynamic flows"*). Given a network and a time horizon T, Single Most Vital Link Over Time (SMVLOT) problem looks for a link whose removal results in greatest decrease in the value of maximum flow over time (*dynamic maximum flow*) up to time horizon T between two terminal nodes. SMVLOT problem is formulated as a mixed binary linear programming problem. This formulation is extended to a general case called k-Most Vital Links Over Time (KMVLOT) problem, in which we look for finding those k links whose removal makes greatest decrease in the value of maximum flow over time. A Benders decomposition algorithm is proposed for solving SMVLOT and KMVLOT problems. For the case of SMVLOT problem, the proposed algorithm is improved to a fully combinatorial algorithm by adopting an iterative method for solving existing integer programming problem. However, our experimental results showed the superiority of proposed methods.

Keywords: Most vital link, Flows over time, Mixed integer programming
AMS Classification: 90C11

1. Introduction

Static (traditional) most vital link problem seeks for a link whose removal from network causes maximum decrease in static (not over time) maximum flow between source node and sink node. Initially Wollmer [17] defined this problem and proposed a solution algorithm for this problem. Wollmer's algorithm is based on the well known and traditional Max-Flow Min-Cut theorem. Corley and Sha [8] extended this problem to find a link whose removal from network results in the greatest increase in shortest distance between two specified nodes. Malik et al. [13] defined k-most vital links on shortest route problem and proposed an algorithm for solving this problem. Their algorithm attempts to find the single and k-most vital links of a network using an iterative labeling procedure which has a time complexity equal to that of Dijkstra's algorithm [1] for the traditional shortest path problem. Lin and Chern [12] extended most vital link problem on fuzzy shortest path problem. Bar-Noy et al. [5] studied complexity of finding k-most vital links and nodes in a network and proved that both of them are NP-hard. Until recent years, static most vital link problem studied as a special case of network interdiction problem (see e.g. [18], [2]).

Although static most vital link problem is studied widely in literature, but none of existing models does not consider the most important role of time factor in real world applications and existing models can not handle with networks in which flows depend on time. For example to determine most vital links in a traffic network, static version of most vital link model is not a subsequent tool, since in this type of networks every link has a traverse time (the time that an auto takes to traverse that link) that static network flows can not deal with such flows. Existing models, which all are

Corresponding Author. Email: ShMorowati@live.com.

based on static network flows, does not consider time-dependent network flows; and as we know in most of real world networks time plays an important role. Providing more realistic model and obviating the deficiency of existing models motivated us to introduce SMVLOT and KMVLOT problems in this paper which take into account

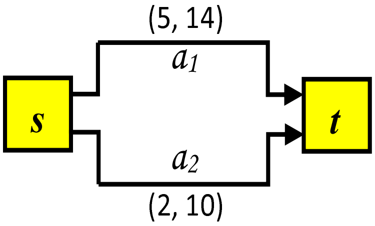

Figure 1. A Simple Network

the time factor in decision making. The model studied in this paper has completely different nature from all of existing models in literature, and none of existing algorithms can be applied to the problem studied in this paper. To formulate SMVLOT and KMVLOT problems and to consider the most important role of time, we use continuous time version of maximum flow over time problem.

Initially Ford and Fulkerson [10] introduced discrete time version of maximum flow over time problem. Since introduction of flows over time the research on this subject has taken two directions. One direction models time in discrete time steps(see e.g. [7], [11]). The other direction models time continuously (see e.g. [9], [16], [3], [4] and [15]). In this paper we consider just continuous time version of this problem and here after we refer to continuous time version of maximum flow over time problem simply by *maximum flow over time problem*. Maximum flow over time problem for time horizon T is defined on a network $G = (N, A, \mathbf{u}, \boldsymbol{\tau}, \{s, t\})$, where N is the set of nodes, A is the set of directed links with a positive capacity $\mathbf{u} = (u_a)_{a\in A}$ and positive transit times $\boldsymbol{\tau} = (\tau_a)_{a\in A}$, s is source node and t is terminal node. Traverse time of the link $a = (i, j) \in A$ is the time that a unit of flow takes to move through this link; more formally, if one unit of flow leaves node i at time θ then one unit of flow arrives node j at time $\theta + \tau_a$. Given G and a time horizon T, the aim of maximum flow over time problem is to find a flow over time pattern which sends maximum value of flow from s to t up to time horizon T.

Given a network and its links capacities and traverse times, the objective of SMVLOT problem is to determine a link whose simultaneous removal from the network causes the greatest decrease in the value of maximum flow over time up to time horizon T between two specified terminal nodes. The traditional most vital link problem and what we study in this paper have completely different nature. To better understanding of SMVLOT problem and to distinguish differences between SMVLOT problem and traditional static most vital link problem, consider the simple example of Figure 1. The 2-tuple vector beside each link shows (u_a, τ_a), where u_a is link capacity and τ_a is link traverse time. Static most vital link problem (withdrawing traverse times) implies that a_1 is the most vital link, because its capacity (i.e. 5) is greater than that of a_2, therefore a_1 carries more flow in a static maximum flow pattern. Note that static maximum flow does not consider the traverse time of links. But to find SMVLOT in this simple network, note that for a given time horizon T, in a maximum flow over time pattern total flow which arrives to node t up to time T from a_1 and a_2 are respectively $(\max\{0, T - \tau_{a_1}\})u_{a_1}$ and $(\max\{0, -\tau_{a_2}\})u_{a_2}$; therefore we must distinguish between following five cases:

a) $0 \le T \le 10$. In this case no flow enters terminal node up to time T since both links traverse times is greater than T, therefore both links can be selected SMVLOT.

b) $10 < T \le 14$. In an optimal maximum flow over time pattern a_1 carries no flow to terminal node up to time T since $(\max\{0, T - \tau_{a_1}\}) = 0$ while a_2 carries a positive flow $(\max\{0, T - \tau_{a_2}\})u_{a_2} = (T-10)2$. As a result in this case a_2 is the SMVLOT, since contribution in the maximum flow up to time T is greater than that of a_1.

c) $14 < T < \frac{50}{3}$. In this case SMVLOT is a_2, because total arrival flow to t from a_1 and a_2 up to time T is equal to $5(T-14)$ and $2(T-10)$, respectively. But as we see, $5(T-14) < 2(T-10)$ for all $T < \frac{50}{3}$.

d) $T = \frac{50}{3}$. Since total flow that arrives to t from a_1 and a_2 up to time $\frac{50}{3}$ is equal, therefore both links can be selected as SMVLOT.

e) $T > \frac{50}{3}$. In this case a_1 is the SMVLOT since total arrival flow to t from a_1 exceeds that of a_2 for all $T > \frac{50}{3}$.

As we see, taking into account the time factor affects our decision. For the simple network of Figure 1 the SMVLOT problem distinguishes between five cases while static most vital link problem determines a_1 as most vital link without taking into account the time factor. Of course in reality the problem is not as simple as that of Figure 1, and real networks may be very complicated. Therefore it seems necessary to formulate and develop an efficient solution method for this problem.

The SMVLOT and KMVLOT problems have many real world application areas such as traffic management, emergency operations (e.g emergency evacuation and emergency dispatching), military operations, smuggling prevention, stadium evacuation, project management, telephone and internet network and so on. Note that traditional most vital link problem may be used to handle with these applications without considering the important role of time factor; but needless to say, in many of these applications such as emergency operations, traffic management and etc. time plays the most important role and withdrawing time factor may deviate right decision making.

The rest of this paper is organized as follows. Section 2 consists of basic notations, definitions which will be used in next sections, and a brief description about maximum flow over time problem. Main contribution of the paper to the literature is presented in Section 3 which consists of mathematical formulation of SMVLOT problem, a basic and an improved Benders decomposition algorithm for solving this problem and several new and original results about SMVLOT problem. In section 4 we have extended mathematical formulation and all results of section 3 to k-most vital links over time problem. In section 5 some experimental results have been provided using proposed solution methods.

2. Preliminaries

2.1. Notation and definitions

Let $G = (N, A, \mathbf{u}, \boldsymbol{\tau}, s, t)$ is given. A route from node s to node t is called an s-t-route and an s-t-route containing link a is called an a-crossing route. Let p be an s-t-route and i, j two nodes on p, the sum of traverse times of all links between nodes i and j which belong to route p is denoted by $\tau_{p(i,j)}$.

A static s-t-flow is a real valued mapping $\mathbf{x}$ on the links of G that satisfies capacity constraints $0 \le x_a \le u_a$ for all $a \in A$ and flow conservation constraints $\sum_{\substack{a\in A\\ a_h=i}} x_a - \sum_{\substack{a\in A\\ a_t=i}} x_a = 0$, $\forall i \in N \setminus \{s,t\}$. The value of a static s-t-flow $\mathbf{x}$ is equal to $|\mathbf{x}| = \sum_{\substack{a\in A\\ a_h=t}} x_a - \sum_{\substack{a\in A\\ a_t=t}} x_a$. A static circulation is a static s-t-flow that must also satisfy conservation constraints at the terminals.

Let the sets of all real numbers and nonnegative real numbers are denoted by $\Re$ and $\Re^+$, respectively. Given G and a time horizon $T \in \Re^+$, a flow over time on G is defined as an array of nonnegative functions such as $\mathbf{f} = (f_a)_{a\in A}$, where for each link $a \in A$, $f_a : \Re \to \Re^+$ is a Lebesgue-integrable function which is zero for all $\theta \in \Re \setminus [0, T - \tau_a)$; in other words, $f_a(\theta) = 0$ must hold except for some $\theta \in [0, T)$. A flow over time $\mathbf{f} = (f_a)_{a\in A}$ is called feasible if it satisfies the capacity constraints $f_a(\theta) \le u_a$, $\forall a \in A, \theta \in [0, T)$. A feasible flow over time $\mathbf{f}$ which satisfies flow conservation constraints

$$\int_0^\theta (\sum_{\substack{a\in A\\ a_h=i}} f_a(\eta - \tau_a) - \sum_{\substack{a\in A\\ a_t=i}} f_a(\eta))d\eta \ge 0, \quad \forall i \in N \setminus \{s,t\}, \theta \in [0,T),$$

$$\int_0^T (\sum_{\substack{a\in A\\ a_h=i}} f_a(\eta - \tau_a) - \sum_{\substack{a\in A\\ a_t=i}} f_a(\eta))d\eta = 0, \quad \forall i \in N \setminus \{s,t\},$$

is called an s-t-flow over time. The value of an s-t-flow over time $\mathbf{f}$ up to time horizon T, is equal to $v_{\mathbf{f}}(T) = \int_0^T (\sum_{\substack{a\in A\\ a_h=t}} f_a(\eta - \tau_a) - \sum_{\substack{a\in A\\ a_t=t}} f_a(\eta))d\eta$.

Let $\mathbf{x}$ be a static s-t-flow in G. Using the renowned flow decomposition theorem [1], decompose $\mathbf{x}$ into some s-t-route and cycle flows $(x_p)_{p\in\mathcal{P}\cup\mathcal{C}}$, where $\mathcal{P} = \{p^1, p^2, \ldots, p^k\}$ and $\mathcal{C} = \{p^{k+1}, p^{k+2}, \ldots, p^{k+z}\}$ are decomposed routes and cycles with respect to $\mathbf{x}$. Then the corresponding *temporally repeated flow* $\mathbf{f}$ for time horizon T is defined as follows

$$f_a(\theta) = \sum_{p^i\in\mathcal{P}(\theta)} x_{p^i}, \forall a \in A, \theta \in [0,T),$$

where $\mathcal{P}(\theta) = \{p^i : i \le k, a \in p^i, \tau_{p^i(s,a_t)} \le \theta, \tau_{p^i(a_t,t)} < T - \theta\}$.

An intuitive interpretation of temporary repeated flow corresponding to static s-t-flow $\mathbf{x}$ is that for each route $p^i \in \mathcal{P}$, send flow at rate x_{p^i} into p^i from the source s during the time interval $[0, T - \tau_{p^i})$ and let the flow progress towards the sink without any delay at intermediate nodes.

All over the paper, we make the following assumption.

Assumption 1. *There exists any unbounded s-t-route p such that $\tau_{p(s,t)} < T$.*

This assumption is not restrictive, since if there exists an s-t-route with a capacity equal to $+\infty$ and traverse time less than T, then the problem has infinite optimal solution. That is, using such route we can sent infinite units of flow from s to t through G up to time T.

2.2. Maximum flow over time problem

Given G and T, the task of maximum flow over time problem is to send as much flow as possible from source node to terminal node up to time horizon T. In other words, maximum flow over time problem seeks an s-t-flow over time which has minimum value among all s-t-flows over time. In continuous time setting, this problem can be formulated as follows[16].

$$\max_{\mathbf{f}} v_{\mathbf{f}}(T)$$
$$\text{s.t.}$$
$$\int_0^{\theta} \Big(\sum_{\substack{a\in A\\ a_h=i}} f_a(\eta-\tau_a) - \sum_{\substack{a\in A\\ a_t=i}} f_a(\eta) \Big) d\eta \geq 0, \quad \forall i \in N\setminus\{s,t\}, \theta\in[0,T), \quad (1)$$
$$\int_0^{T} \Big(\sum_{\substack{a\in A\\ a_h=i}} f_a(\eta-\tau_a) - \sum_{\substack{a\in A\\ a_t=i}} f_a(\eta) \Big) d\eta = \begin{cases} -v_{\mathbf{f}}(T) & i=s \\ 0 & \forall i\in N\setminus\{s,t\} \\ v_{\mathbf{f}}(T) & i=t, \end{cases} \quad (2)$$
$$0 \leq f_a(\theta) \leq u_a, \qquad \forall a\in A, \theta\in[0,T). \quad (3)$$

The optimal solution for maximum flow over time problem and its optimal value is denoted by $\mathbf{f}^*$ and $v^*(T)$, respectively.

Ford and Fulkerson [10] showed that if $\mathbf{x}$ be such a static s-t-flow that maximizes $-T|\mathbf{x}| + \sum_{a\in A}\tau_a x_a$, then temporary repeated flow corresponding to $\mathbf{x}$, is an optimal solution for discrete time version of maximum flow over time problem. As a generalization to results of Ford and Fulkerson [10] to continuous-time setting, by following theorem Anderson and Philpott [4] showed that the optimum value of complicated problem (1) can be obtained by solving a static minimum cost circulation problem which is polynomially solvable.

Theorem 1. *(Anderson and Philpott [4]). The continuous maximum flow over time up to time horizon T in the network $G=(N,A,\mathbf{u},\boldsymbol{\tau},s,t)$ has value $v^*(T) = Tx^*_{ts} - \sum_{a\in A}\tau^*_a x^*_a$, , where $\mathbf{x}^*$ is a minimum cost circulation in the network with an additional sink-to-source link with cost $-T$ and infinite capacity. (x^*_{ts} denotes the flow on artificial link (t,s)).*

This theorem declares that the value of maximum flow over time up to time horizon T is equal to optimal objective value of following minimum cost circulation problem which is defined on network G with an additional link (t,s) with cost $-T$ and infinite capacity.

$$\min_{\mathbf{x}} -Tx_{ts} + \sum_{a\in A}\tau_a x_a$$
$$\text{s.t.} \quad \sum_{\substack{a\in A\\ a_h=i}} x_a - \sum_{\substack{a\in A\\ a_t=i}} x_a = 0, \qquad \forall i\in N, \quad (4)$$
$$x_a \leq u_a, \qquad \forall a\in A, \quad (5)$$
$$x_a \geq 0, \qquad \forall a\in A', \quad (6)$$

where $A' = A\cup\{(t,s)\}$. Notice that according to Assumption 1 feasible region of above circulation problem is a closed convex set. For future purposes we reformulate this minimization problem as following maximization problem.

$$\max_{\mathbf{x}} Tx_{ts} - \sum_{a\in A}\tau_a x_a \quad (7)$$
$$\text{s.t.} \quad (4)-(6).$$

The temporary repeated flow corresponding to optimal solution of (7), provides an optimal solution for (1).

3. Single Most Vital Link Over Time Problem

Theoretically, the SMVLOT problem and KMVLOT problem are the same. Therefore to avoid confusing and more notations, in this section we just study in detail the SMVLOT problem as a special case of KMVLOT problem; and in the next section all results addressed in this section may be simply extended to global case of KMVLOT. In this section we attempt to formulate SMVLOT problem mathematically, afterwards we will provide a solution method for this problem and finally we prove some results in order to clarify some properties of SMVLOT problem. First of all we recall the SMVLOT problem using a notational language.

Given a network structure $G=(N,A,\mathbf{u},\boldsymbol{\tau},s,t)$, let G_a denotes a network same as G in which link a is blocked. To block a link means that no flow can traverse through corresponding link. Denote the value of maximum flow over time between terminal nodes $\{s,t\}$ up to time horizon T in networks G and G_a by $M(T)$ and $M_a(T)$, respectively. Using these notations, the single most vital link of G corresponding to time horizon T is a link with maximum value of $M(T)-M_a(T)$(i.e. minimum value of $M_a(T)$) among all links $a\in A$. Note that this definition for SMVLOT is consistent with our previous definition as a link whose blockage causes the greatest decrease in the value of maximum flow over time up to time horizon T between terminal nodes $\{s, t\}$. Generally the single most vital link in G for time horizon T is not unique. Taking into account this observation, let $a_v(T)$ denote a SMVLOT in network G for time horizon T and $A_v(T)$ denote the set of all single most vital links for time horizon T. We denote by $\mathbf{e}_a$ an $|A|-$tuple vector with 1 as the element corresponding to link a and 0 otherwise. By this introduction we develop a mathematical model for SMVLOT problem.

3.1. Mathematical formulation

To formulate SMVLOT problem we define a set of binary variables ϕ_a assigned to each link $a \in A$. We mean by $\phi_a = 1$ that link a is blocked and otherwise link a is not blocked($\phi_a = 0$). Using these considerations, let Φ be the set of all possible elections for the single most vital link over time in G; that is $\Phi = \{\boldsymbol{\phi} \in \{0,1\}^{|A|} : \sum_{a\in A} \phi_a = 1\}$. As is obvious, Φ is the set of all vectors $\mathbf{e}_{a_1}, \mathbf{e}_{a_2}, \cdots, \mathbf{e}_{a_{|A|}}$. To block a link a in mathematical model, we can simply increase its traverse time to a number greater than T, because if the traverse time of a link be grater than T then the traverse time of every a-crossing route will be greater than T, therefore in a maximum flow over time pattern no flow arrives to t from such routes up to time horizon T. In this case, this implies that link a will not be used in any maximum flow over time pattern. Given $\boldsymbol{\phi} \in \Phi$, let a be those link for which $\phi_a = 1$ and denote the set of all s-t-flows over time in G_a by $F(\boldsymbol{\phi})$. In other words, $F(\boldsymbol{\phi})$ contains the set of all $\mathbf{f} = (f_a(\theta))_{a\in A}$ that satisfies following relations in G.

$$\int_0^\theta [\sum_{\substack{a\in A \\ a_h=i}} f_a(\eta - (\tau_a + T\phi_a)) -$$

$$\sum_{\substack{a\in A \\ a_t=i}} f_a(\eta)] d\eta \geq 0, \forall i \in N \setminus \{s,t\}, \theta \in [0,T),$$

$$\int_0^T [\sum_{\substack{a\in A \\ a_h=i}} f_a(\eta - (\tau_a + T\phi_a)) - \sum_{\substack{a\in A \\ a_t=i}} f_a(\eta)] d\eta$$

$$= \begin{cases} -v_{\mathbf{f}}(T) & i = s \\ 0 & i \in N \setminus \{s,t\} \\ v_{\mathbf{f}}(T) & i = t, \end{cases}$$

$$0 \leq f_a(\theta) \leq u_a, \qquad \forall a \in A, \theta \in [0,T).$$

As is stated in the beginning of Section 3, the SMVLOT may be defined as

$$\text{argmin}_{a\in A}\{M_a; a \in A\}.$$

In other words, $a \in A_v(T)$ if the value of maximum flow over time up to time horizon T in G_a be less than that of all other links in A. Denoting the value of maximum flow over time for all $\mathbf{f} \in F(\boldsymbol{\phi})$ by $H(\boldsymbol{\phi})$, since each $\boldsymbol{\phi} \in \Phi$ is equivalent with blocking one of links in G, therefore the SMVLOT problem looks for a $\boldsymbol{\phi} \in \Phi$ for which $H(\boldsymbol{\phi})$ is minimized.

Using these notations and introduction we can formulate most single vital link over time problem as following mini-max problem.

$$\begin{aligned} &\min_{\boldsymbol{\phi}\in\Phi} H(\boldsymbol{\phi}) \\ &H(\boldsymbol{\phi}) = \max_{\mathbf{f}} v_{\mathbf{f}}(T) \\ &\text{s.t.} \qquad \mathbf{f} \in F(\boldsymbol{\phi}). \end{aligned} \tag{8}$$

Note that (8) is a very complicated problem which can not be solved directly and we must do some reformulations for this problem to provide a solution method.

According to Theorem 1, for a fixed and constant $\boldsymbol{\phi} \in \Phi$, optimum value of inner maximum flow over time problem in (8) is equal to optimum value of following circulation problem.

$$\mathbf{MCCP}(\boldsymbol{\phi}, T): \qquad \max_{\mathbf{x}} T x_{ts} - \sum_{a\in A} (\tau_a + T\phi_a) x_a \\ \text{s.t.} \quad (4)-(6).$$

Notice that for a fixed $\boldsymbol{\phi}$, if $\phi_a = 1$ then the penalized traverse time of link a is $\tau_a + T$. As a result the traverse time of every a-crossing route is greater than T. Therefore every positive flow on such routes decreases objective function of MCCP$(\boldsymbol{\phi}, T)$ since, the contribution of an $\epsilon > 0$ flow on an $a-$crossing route p in objective function of MCCP$(\boldsymbol{\phi}, T)$ is $(T - \tau_p)\epsilon$ which is less than zero. This implies that in optimal solution $x_a = 0$ must hold in an optimal solution, since x_a is the sum of flows on all a-crossing routes. The same discussion on (8) implies that if $\phi_a = 1$ then in an optimal flow over time pattern $f_a(\theta) = 0$ for all $\theta \in [0, T)$. Using Theorem 1, since $H(\phi)$ equals optimum value of MCCP$(\boldsymbol{\phi}, T)$, then (8) may be reformulated as following mini-max problem.

$$\begin{aligned} &\min_{\boldsymbol{\phi}\in\Phi} H(\boldsymbol{\phi}) \\ &H(\boldsymbol{\phi}) = \max_{\mathbf{x}} T x_{ts} - \sum_{a\in A} (\tau_a + T\phi_a) x_a \\ &\text{s.t.} \qquad (4)-(6). \end{aligned} \tag{9}$$

Now, for fixed $\boldsymbol{\phi}$ according to strong duality theorem, taking dual of inner maximization problem in (9) gives following equivalent model.

$$\begin{aligned}
&\min_{\boldsymbol{\phi}\in\Phi} H(\boldsymbol{\phi})\\
&H(\boldsymbol{\phi}) = \min_{\boldsymbol{\alpha},\boldsymbol{\mu}} \sum_{a\in A} u_a \mu_a\\
&\text{s.t.} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (10)\\
&\mu_a + \alpha_{a_t} - \alpha_{a_h} + T\phi_a \geq -\tau_a, \qquad \forall a \in A,\\
&\alpha_t - \alpha_s \geq T,\\
&\mu_a \geq 0, \quad \phi_a \in \{0,1\}, \qquad\qquad \forall a \in A.
\end{aligned}$$

Now by releasing $\boldsymbol{\phi}$, we can transform (10) to following mixed linear integer programming problem.

$$\begin{aligned}
&\min_{\boldsymbol{\phi},\boldsymbol{\alpha},\boldsymbol{\mu}} \sum_{a\in A} u_a \mu_a\\
&\text{s.t.}\\
&\mu_a + \alpha_{a_t} - \alpha_{a_h} + T\phi_a \geq -\tau_a, \quad \forall a \in A, \quad (11)\\
&\alpha_t - \alpha_s \geq T,\\
&\sum_{a\in A} \phi_a = 1,\\
&\mu_a \geq 0, \quad \phi_a \in \{0,1\}, \qquad\qquad \forall a \in A.
\end{aligned}$$

We have transformed the complicated problem (8) into the mixed linear minimization problem (11) which is solvable by all existing methods for solving mixed linear programming problem.

Although (11) can be solved by well-known methods in literature such as branch and bound methods or cutting plane methods but our experiments demonstrated that, when the size of network grows, applying branch and bound methods directly to the problem (11), is time consuming (see section 5; in some cases optimal solution is achieved after performing millions of iterations). To provide a computationally efficient and a fast solution method, according to special structure of (11) we propose two Benders decomposition type algorithms [6] to this problem. The advantage of proposed algorithms versus branch and bound methods is their combinatorial or semi-combinatorial nature. However, one of the proposed algorithms is semi-combinatorial while the other one is fully combinatorial.

3.2. A basic benders decomposition algorithm

In this section we propose a Benders decomposition type algorithm [6] to find $a_v(T)$ for given G and time horizon T. Generally benders decomposition algorithm is a suitable solution method for those problems containing two groups of variables with different nature. In the case of SMVLOT problem since all variables of its equivalent problem (11) can be decomposed into two groups (i.e. binary variable $\boldsymbol{\phi}$ and continuous variables $\boldsymbol{\mu}$ and $\boldsymbol{\alpha}$) and the feasible region of its dual (i.e. MCCP$(\boldsymbol{\phi}, T)$) does not depend on $\boldsymbol{\phi}$, therefore Benders decomposition algorithm [6] is a suitable tool for solving (11). More formally, note that (11) may be rewritten as

$$\min_{\boldsymbol{\phi}\in\Phi} \left\{ \min \left\{ \sum_{a\in A} u_a \mu_a : \begin{array}{l} \boldsymbol{\mu}, \boldsymbol{\alpha} \text{ satisfy the} \\ \text{constraints of(11)} \end{array} \right\} \right\} (12)$$

Now for a fixed $\boldsymbol{\phi} \in \Phi$, taking dual of inner minimization problem with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\alpha}$ implies the following reformulation for (12).

$$\min_{\boldsymbol{\phi}\in\Phi} \{\max\{Tx_{ts} - \sum_{a\in A} (\tau_a + T\phi_a) : (4) - (6)\}\} \quad (13)$$

Notice that the feasible region of maximization problem in (13), which is equivalent with that of MCCP$(\boldsymbol{\phi}, T)$, is nonempty ($\mathbf{x}=\mathbf{0}$ is feasible) and does not depend on $\boldsymbol{\phi}$. According to Assumption 1, the inner maximization problem in (13) has also a finite optimal solution, therefore according to duality theory its dual is feasible and has a finite optimal solution equal with that of maximization problem in (13). Using this discussion let X denote the set of all extreme points of feasible region of inner maximization problem in (13), then according to the well known property of linear programming, X consists the optimal solution of inner maximization problem in (13). As a result of this explanation (13) is equivalent to

$$\begin{aligned}
&\min_{\boldsymbol{\phi}\in\Phi} q \qquad\qquad\qquad\qquad\qquad\qquad\qquad (14)\\
&\text{s.t.} \quad Tx_{ts} - \sum_{a\in A} (\tau_a + T\phi_a) x_a \leq q; \forall \mathbf{x} \in X.
\end{aligned}$$

But in practice finding all extreme points (i.e X) is impossible; therefore Benders algorithm decomposes the problem into a master problem and a sub-problem and instead of X this decomposition uses just a subsequence of Xsuch as $\hat{X}$. However, Benders decomposition algorithm updates $\hat{X}$ in each iteration by adding a new extreme point. A Benders decomposition for (13) may be formulated as follows.

$$[\text{Master}(\hat{X})] \qquad \min_{\phi \in \Phi} q$$

$$\text{s.t.} \quad Tx_{ts} - \sum_{a \in A} (\tau_a + T\phi_a) x_a \le q; \forall \mathbf{x} \in \hat{X} \quad (15)$$

$$[\text{Sub}(\boldsymbol{\phi})] \qquad \max_{\mathbf{x}} Tx_{ts} - \sum_{a \in A} (\tau_a + T\phi_a) x_a$$

$$\text{s.t.} \qquad (4) - (6).$$

In each iteration the algorithm solves Sub($\boldsymbol{\phi}$) and updates $\hat{X}$ by adding a new extreme point and then Master($\hat{X}$) seeks for the suboptimal $\boldsymbol{\phi}$ to improve previous $\boldsymbol{\phi}$ by examining all $\mathbf{x}$ in updated $\hat{X}$. Note that in each iteration Master($\hat{X}$) provides a lower bound and Sub($\boldsymbol{\phi}$) provides an upper bound on optimal solution of original problem. The algorithm terminates when upper bound and lower bound be equivalent. Although it is possible that the algorithm finds optimal solution when $\hat{X} = X$, but, as experimental results showed, we hope to find optimal solution generating only a small fraction of extreme points. By this discussion, basic Benders decomposition algorithm can now be stated as follows.

Basic Benders Decomposition Algorithm

Input:	$G = (N, A, \mathbf{u}, \boldsymbol{\tau}, s, t)$ and T.
Output:	Single most vital link of G corresponding to T and $M_{a_v(T)}(T)$.
Step 0.	UB $\leftarrow +\infty$; LB $\leftarrow -\infty$; $\hat{X} \leftarrow \emptyset$. $\hat{\boldsymbol{\phi}} \leftarrow \mathbf{0}$.
Step 1.	Solve Sub($\hat{\boldsymbol{\phi}}$) to obtain optimal solution $\mathbf{x}^*(\hat{\boldsymbol{\phi}})$.
Step 2.	$\hat{X} \leftarrow \hat{X} \cup \mathbf{x}^*(\hat{\boldsymbol{\phi}})$. $s \leftarrow Tx^*_{ts}(\hat{\boldsymbol{\phi}}) - \sum_{a \in A} \tau_a x^*_a(\hat{\boldsymbol{\phi}})$.
Step 3.	If s <UB then UB $\leftarrow s$ and $\boldsymbol{\phi}^* \leftarrow \hat{\boldsymbol{\phi}}$.
Step 4.	If UB = LB then STOP. $\boldsymbol{\phi}^*$defines the most vital link. Otherwise, go to Step 5.
Step 5.	Solve Master($\hat{X}$) to find $\boldsymbol{\phi}^*$ and q^*; LB $\leftarrow q^*$.
Step 6.	If UB = LB then STOP. $\boldsymbol{\phi}^*$defines the most vital link. Otherwise, $\hat{\boldsymbol{\phi}} \leftarrow \boldsymbol{\phi}^*$; and go to Step 1.

Note that, Sub($\boldsymbol{\phi}$) in Step 1 of the algorithm is a minimum cost circulation problem for which there exist polynomial time solution algorithms (see [1, chap. 9]) and in Step 5 Master($\hat{X}$) is a binary programming problem which can be solved using well known branch and bound method.

The correctness of the algorithm, as in any Benders decomposition algorithm, is based on the following observations:

1. The subproblem finds an optimal follower's response for selected single most vital link. Hence, optimal solution of Sub($\boldsymbol{\phi}$) gives an upper bound on the leader's optimal objective value.
2. When $\hat{X} \subseteq X$, Master($\hat{X}$) is a relaxation of (9) and optimum value of Master($\hat{X}$) is a lower bound on the leader's optimal objective value.
3. If the subproblem produced same solution twice, the upper and lower bounds match and the algorithm terminates. The algorithm converges after finite iterations because the number of extreme points of feasible region of Sub($\boldsymbol{\phi}$) is finite.

Since that master problem in Benders decomposition algorithm is an integer programming problem and the minimum cost circulation problem (i.e. Sub($\boldsymbol{\phi}$)) can be solved using existing polynomial time and fully combinatorial algorithms ([1]), then the derived basic benders decomposition algorithm has a semi-combinatorial nature vs branch and bound method which has completely non-combinatorial nature.

The algorithm may suffer from solving a binary integer programming problem in Step 5 of each iteration by branch and bound methods. To avoid the complexity of solving integer programming problem analytically, an improved Benders decomposition algorithm is proposed in the next section which uses an iterative procedure for solving master problem in basic Benders decomposition algorithm. This improvement transforms the semi-combinatorial basic benders decomposition algorithm to a fully combinatorial algorithm which we will present in the next section.

3.3. An improved benders decomposition algorithm

3.3.1. The algorithm

Solving the binary programming problem in Step 5 of basic Benders decomposition algorithm may be problematic and time consuming when $\hat{X}$ grows in large scale networks. According to special structure of Φ, solving Master($\hat{X}$) by an iterative procedure, the difficulty of solving Master($\hat{X}$) in basic Benders decomposition algorithm by analytical methods may be driven away.

Improved Benders Decomposition Algorithm

Input:	$G = (N, A, \mathbf{u}, \boldsymbol{\tau}, s, t)$ and T.
Output:	Single most vital link of G corresponding to T and $M_{a_v(T)}(T)$.
Step 0.	UB $\leftarrow +\infty$; LB $\leftarrow -\infty$; $\hat{X} \leftarrow \emptyset$. $\hat{\boldsymbol{\phi}} \leftarrow \mathbf{0}$; $\mathbf{z} \leftarrow -\infty^{\lvert A \rvert}$.
Step 1.	Solve MCCP($\hat{\boldsymbol{\phi}}, T$) to obtain optimal solution $\mathbf{x}^*(\hat{\boldsymbol{\phi}})$.
Step 2.	$\hat{X} \leftarrow \hat{X} \cup \mathbf{x}^*(\hat{\boldsymbol{\phi}})$; $s \leftarrow Tx^*_{ts}(\hat{\boldsymbol{\phi}}) - \sum_{a \in A} \tau_a x^*_a(\hat{\boldsymbol{\phi}})$.
Step 3.	If $s <$ UB then UB $\leftarrow s$ and $\boldsymbol{\phi}^* \leftarrow \hat{\boldsymbol{\phi}}$.
Step 4.	If UB = LB then STOP. $\boldsymbol{\phi}^*$defines the most vital link. Otherwise, go to Step 5.
Step 5.	For all $a \in A$, if $-Tx^*_a(\hat{\boldsymbol{\phi}}) + s > z_a$ then $z_a \leftarrow -Tx^*_a(\hat{\boldsymbol{\phi}}) + s$.
Step 6.	Select a link a' such that $z_{a'} = \min_{a \in A}\{z_a\}$; LB $\leftarrow z_{a'}$.
Step 7.	If UB=LB then STOP; a' is the most vital link. Otherwise,
Step 8.	$\hat{\boldsymbol{\phi}} \leftarrow \mathbf{e}_{a'}$; and go to Step 1.

In improved Benders decomposition algorithm we have replaced Step 5 of basic Benders decomposition algorithm by Step 5 and Step 6 in new algorithm. The improved algorithm solves no integer linear programming problem directly. Instead of solving master problem in basic Benders decomposition algorithm by analytical methods such as branch and bound methods or cutting plane methods, the improved algorithm uses an iterative procedure for solving Master($\hat{X}$).

As is seen, the improved algorithm solves no integer programming problem directly by analytical methods and instead it solves the binary linear programming of Master($\hat{X}$) using an iterative procedure. From this point of view, the improved algorithm is a fully combinatorial algorithm.

3.3.2. Algorithm correctness

The improved algorithm is the same as basic Benders decomposition algorithm in which Step 5 of basic Benders decomposition algorithm is replaced by Step 5 and Step 6 in improved algorithm. Since correctness of basic Benders decomposition algorithm is demonstrated therefore to prove correctness of improved algorithm it is sufficient to show that Step 5 and Step 6 of improved algorithm is equivalent to Step 5 of basic Benders decomposition algorithm which we have proved in following theorem.

Theorem 2. *The Step 5 and Step 6 of improved algorithm is equivalent with Step 5 of basic Benders decomposition algorithm; that is, Step 5 and Step 6 of improved algorithm solves Master($\hat{X}$) correctly.*

Proof. Consider that we are in iteration k and $\hat{X}$ contains $\mathbf{x}^1, \mathbf{x}^2, \cdots, \mathbf{x}^k$. After Step 5 of improved algorithm for each $\bar{a} \in A$, $z_{\bar{a}}$ is equal to

$$\max\{Tx^i_{ts} - \sum_{\substack{a \in A;\\ a \neq \bar{a}}} \tau_a x^i_a - (\tau_{\bar{a}} + T)x_{\bar{a}} : \mathbf{x}^i \in \hat{X}\}.$$

Step 6 selects link a' which has minimum value of $z_{a'}$; Notice that $z_{a'}$ is minimum value which $Tx^i_{ts} - \sum_{\substack{a \in A;\\ a \neq a'}} \tau_a x^i_a - (\tau_{a'} + T)x_{a'} \leq z_{a'}$ holds for all $\mathbf{x}^i \in \hat{X}$. Since Master($\hat{X}$) seeks minimal q^* such that $Tx^i_{ts} - \sum_{\substack{a \in A;\\ a \neq a'}} \tau_a x^i_a - (\tau_{a'} + T)x_{a'} \leq q^*$ hold for all $\mathbf{x}^i \in \hat{X}$; this implies that $z_{a'} = q^*$. $\square$

Theorem 2 implies that Step 5 and Step 6 of improved algorithm solves Master problem in basic Benders decomposition algorithm correctly. As a result the improved algorithm solves (11) correctly.

3.4. Several results

In this section we prove several properties of SMVLOT problem which will be helpful in practice. Consider a case in which we want to determine $a_v(T)$ and $M_{a_v(T)}(T)$ for all $T \in (0, \hat{T})$. As we know this is practically impossible specially for large scale networks and big values of $\hat{T}$. In this section we prove that the function $M_{a_v(T)}(T)$ is a continuous, piecewise linear and quasi-convex function which makes it possible to determine $a_v(T)$ and $M_{a_v(T)}(T)$ for all $T \in (0, \hat{T})$ just by determining them in finite number of time horizons $T_1, T_2, ..., T_n$ which will be referred as *"critical time horizons"*.

Another result which we prove in this section demonstrates that there exists a time horizon that $A_v(T)$ remains unchanged for greater time horizons. This emphasizes that the sequence of critical time horizons is finite therefore if we find the greatest critical time horizon then we can obtain $A_v(T)$ and $M_{a_v(T)}(T)$ for all time horizons greater than the greatest critical time horizon. Using this property we can determine $a_v(T)$ and $M_{a_v(T)}(T)$ for all $T \in (0, +\infty)$ just by determining them in finite number of time horizons. By this glancing clarification about what we want to demonstrate in this section, we are now ready to discuss about stated results.

Let S be an open interval of $\Re$(real numbers), then we refer to the function $f : S \to \Re$ as a quasi-convex function if for each p and $q \in S$ and for all $\lambda : 0 < \lambda < 1$, f satisfies following inequality.

$$f(\lambda p + (1 - \lambda)q) \leq \max\{f(p), f(q)\}.$$

Following lemma demonstrates that minimum of several increasing, continuous, piecewise linear and convex functions is an increasing continuous, piecewise linear and quasi-convex function.

Lemma 1. *Let $g_1, g_2, \cdots, g_m$ to be m (strictly)increasing continuous, piecewise linear and convex functions from S to $\Re$ and define g as the minimum of these functions in S, i.e.*

$$g(x) = \min_i g_i(x), \qquad \forall x \in S.$$

Then g is an (strictly)increasing, continuous, piecewise linear and quasi-convex function.

Proof. Continuity, increasing and piecewise linear property of g is obvious. We just survey quasi-convexity of g. Let p and $q \in S$, we show that $g(\lambda p + (1 - \lambda)q) \leq \max\{g(p), g(q)\}$ holds for all $0 < \lambda < 1$. Without loss of generality consider that $p < q$. Since g is an increasing function, then $g(p) \leq g(q)$. Therefore $\max\{g(p), g(q)\} = g(q)$. As a result to prove quasi-convexity of g it suffices to show that $g(\lambda p + (1 - \lambda)q) \leq g(q)$ holds for all $0 < \lambda < 1$. Since for all $j \in \{1, 2, \cdots, m\}$, g_j is monotone and convex, therefore

$$\begin{aligned} g(\lambda p + (1 - \lambda)q) &= \min_i\{g_i(\lambda p + (1 - \lambda)q) : \\ &\quad\; i = 1, 2, \cdots, m\} \\ &\leq g_j(\lambda p + (1 - \lambda)q) \\ &\leq \lambda g_j(p) + (1 - \lambda)g_j(q) \\ &\leq \lambda g_j(q) + (1 - \lambda)g_j(q) \\ &\leq g_j(q). \end{aligned}$$

Taking minimum of right hand side over j implies that $g(\lambda p + (1 - \lambda)q) \leq g(q)$. This completes the proof. □

Let T_0 and T_{0a} denote the length of shortest s-t-route in G and G_a, respectively. Using these notations and denoting positive reals by $\Re^+$, we have following result.

Theorem 3. *Given a network $G = (N, A, \mathbf{u}, \boldsymbol{\tau}, s, t)$ and a time horizon T, the function $f : \Re^+ \to \Re^+$, defined by $f(T) = M_{a_v(T)}(T)$, is an increasing, piecewise linear, continuous and quasi-convex function. Moreover for $T \geq T_{0\max}$, $f(T)$ is strictly increasing, where $T_{0\max} = \max_a\{T_{0a} | a \in A\}$.*

Proof. From non-combinatorial view point, for fixed $\boldsymbol{\phi} \in \Phi$, regarding T as a parameter in MCCP$(\boldsymbol{\phi}, T)$ yields a special parametric cost linear programming problem. The optimum value of such parametric maximization linear programming problem is a continuous, piecewise linear and convex function with respect to T (see e.g. Murty[14]). Therefore for each link $a \in A$, $M_a(T)$ is a continuous, piecewise linear and convex function with respect to T. From a combinatorial view point, let $\mathbf{x}^{*aT}$ denote optimal solution of circulation problem MCCP$(\mathbf{e}_a, T)$. If $T \geq T_{0a}$ then x^{*aT}_{ts} will be greater than zero. Notice that for $0 \leq T \leq T_{0a}$, $\mathbf{x}^{*aT} = \mathbf{0}$ is an optimal solution for MCCP$(\mathbf{e}_a, T)$. Since the slope of the piecewise linear function $M_a(T)$ is $x^{*aT}_{ts} \geq 0$, therefore $M_a(T)$ is an increasing function of T. Specially, for $T > T_{0a}$, $x^{*aT}_{ts} > 0$ therefore $M_a(T)$ is strictly increasing for $T \geq T_{0a}$. But notice that for each T, $M_{a_v(T)}(T) = \min_{a \in A}\{M_a(T) | a \in A\}$, therefore according to Lemma 1, $f(T) = M_{a_v(T)}(T)$ is an increasing, continuous, piecewise linear and quasi-convex function. Specially, since all functions $\{M_a(T) | a \in A\}$ are strictly increasing for $T \geq T_{0\max} = \max_a\{T_{0a} | a \in A\}$, then $f(T) = M_{a_v(T)}(T)$ is strictly increasing for all $T \geq T_{0\max}$. □

According to Theorem 3, $M_{a_v(T)}(T)$ is a piecewise linear and quasi-convex function and since $M_{a_v(T)}(T) = \min_{a \in A}\{M_a(T) | a \in A\}$ then each linear piece of $M_{a_v(T)}(T)$ coincides with some of functions $M_a(T)$. As a result following corollary states that the time interval (0,T) can be partitioned into some disjoint intervals such that $A_v(T)$ remains unchanged within each of these disjoint intervals.

Corollary 1. *Given $G = (N, A, \mathbf{u}, \boldsymbol{\tau}, s, t)$ and a time horizon T, there exist $0 = T_0 < T_1 < T_2 < T_3 < \cdots < T_{n-1} < T_n = T$ such that for each $i = 1, 2, \cdots, n$ and for all $\tilde{T}, \hat{T}$ in time interval (T_i, T_{i+1}) we have $A_v(\hat{T}) = A_v(\tilde{T})$.*

Given a time horizon $\hat{T}$ and $A_v(\hat{T})$, let $\bar{T}$ be the greatest time horizon that $A_v(T) = A_v(\hat{T})$ holds for all T in time interval $(\hat{T}, \bar{T})$; we refer to such time horizon(i.e. $\bar{T}$) as *critical time horizon.*

In the following it will be shown that the set of all critical time horizons within time interval $(0, +\infty)$ is a finite set, and as result there exists a time horizon that for greater time horizons $A_v(T)$ remains unchanged.

Proposition 1. *The function $f(T) = M_{a_v(T)}(T)$ is the minimum of functions $\{M_a(T) | a \in A\}$; i.e.*

$$M_{a_v(T)}(T) = \min_{a \in A}\{M_a(T)|a \in A\} \qquad \forall T.$$

Since all functions $M_a(T)$ are increasing, continuous, piecewise linear and convex, then for each critical time horizon $\bar{T}$, there exists at least a pair of links a and a' which $M_a(T)$ crosses $M_{a'}(T)$ at $\bar{T}$. We refer to a time horizon as crossing point *if there exist at least two links a, $a' \in A$ such that $M_{a'}(T)$ and $M_a(T)$ crosses each other at this time horizon. By this definition it can be implied that the set of critical time horizons is a subset of all crossing point.*

Theorem 4. *Given a network $G = (N, A, \mathbf{u}, \boldsymbol{\tau}, s, t)$ there exist a time horizon $\tilde{T}$ such that $A_v(T) = A_v(\tilde{T})$ holds for all $T > \tilde{T}$.*

Proof. For each $a \in A$, $M_a(T)$ is a continuous, increasing, piecewise linear and convex function. In each break point of $M_a(T)$, optimal solutions set of MCCP($\mathbf{e}_a, T$) changes (see Murty[14]). According to Assumption 1, feasible region of MCCP($\mathbf{e}_a, T$) is a closed convex set and since its feasible region does not depend on T then the set of extreme points of MCCP($\mathbf{e}_a, T$) is finite which implies that the number of break points of $M_a(T)$ is finite. As a result there exist a time horizon T_a that $M_a(T)$ is linear for time horizons greater than T_a; in other words $M_a(T)$ has no break point for $T \geq T_a$. Let $T_{max} = \max_a\{T_a|a \in A\}$, then for each $a \in A$, $M_a(T)$ is linear for $T \geq T_{max}$. This implies that the set of functions $\{M_a(T)|a \in A\}$ cross each other in finite number of points for $T \geq T_{max}$. Specially there exists a time horizon $\tilde{T}$ and a link a' which $M_{a'}(T) = \min\{M_a(T) : a \in A\}$ for all $T \geq \tilde{T}$, and $M_{a'}(T)$ does not crosses none of functions $\{M_a(T) : \forall a \in A, a \neq a'\}$. According to Proposition 1, this implies that there exists a critical time horizon that $A_v(T)$ remains unchanged for all $T \geq \tilde{T}$. This completes the proof. $\square$

4. k-Most Vital Links Over Time Problem

We have discussed in detail the SMVLOT problem in the previous section. The SMVLOT problem is a special case of KMVLOT problem with $k = 1$. In this section all of demonstrated results in Section 3 will be generalized to KMVLOT problem. Given G and T, the k-most vital links of G corresponding to T are those k links whose removal from the network results the greatest decrease in value of maximum flow over time through network up to time horizon T. The model (11) can be extended to find k most vital links over time as follows.

$$\begin{aligned}
&\min_{\boldsymbol{\phi},\boldsymbol{\alpha},\boldsymbol{\mu}} \sum_{a \in A} u_a \mu_a \\
&\text{s.t.} \\
&\mu_a + \alpha_{a_t} - \alpha_{a_h} + T\phi_a \geq -\tau_a, \forall a \in A, \qquad (16)\\
&\alpha_t - \alpha_s \geq T, \\
&\sum_{a \in A} \phi_a = k, \\
&\mu_a \geq 0, \quad \phi_a \in \{0, 1\}, \qquad \forall a \in A.
\end{aligned}$$

Let $\Phi(k) = \{\phi \in \{0,1\}^{|A|} : \sum_{a \in A} \phi_a = k\}$, replacing Φ by $\Phi(k)$ in (15), a basic Benders decomposition algorithm can be extended for solving (16).

Some results same as that of Theorem 3, Theorem 4 and Corollary 1 hold for k-most vital links over time problem. In other words, there exists a time horizon $\tilde{T}$ that for time horizons greater than $\tilde{T}$, k most vital links of network does not change over time; and $M(T, k)$ is an increasing, continuous, piecewise linear and quasi-convex function with respect to T, where $M(T, k)$ denotes the value of maximum flow that can be sent from s to t up to time horizon T through network G in which k most vital links corresponding to time horizon T are blocked. These can be proved simply same as proof of Theorem 4 and Theorem 3.

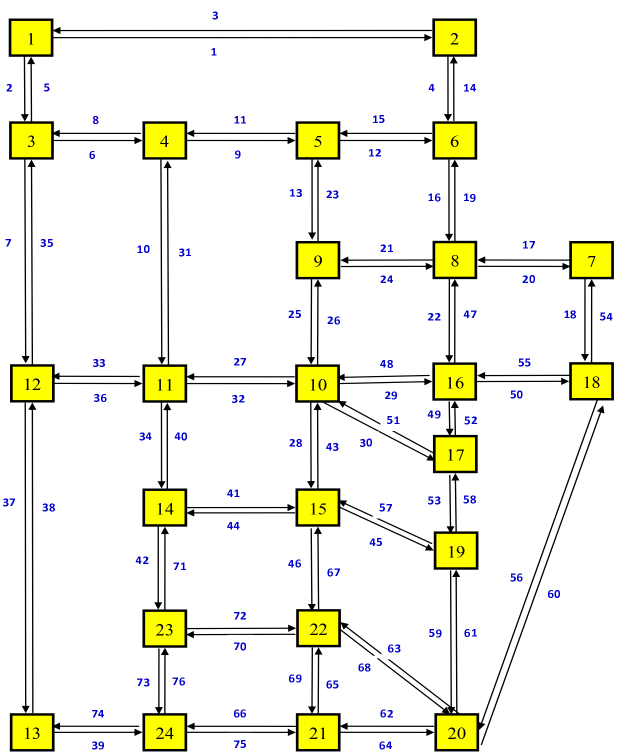

Figure 2. Sioux-Falls Network

Table 1. Each of three column has three sub-columns, first sub-column denotes link ID, second one the capacity of link and third one traverse time of corresponding link.

ID	u	τ	ID	u	τ	ID	u	τ
1	25.9002	3.6	27	10	3	53	4.8240	1.2
2	23.4035	2.4	28	13.5120	3.6	54	23.4035	1.2
3	25.9002	3.6	29	4.8549	2.4	55	19.6799	1.8
4	4.9582	3	30	4.9935	4.8	56	23.4035	2.4
5	23.4035	2.4	31	4.9088	3.6	57	14.5648	1.8
6	17.1105	2.4	32	10	3	58	4.8240	1.2
7	23.4035	2.4	33	4.9088	3.6	59	5.0026	2.4
8	17.1105	2.4	34	4.8765	2.4	60	23.4035	2.4
9	17.7828	1.2	35	23.4035	2.4	61	5.0026	2.4
10	4.9088	3.6	36	4.9088	3.6	62	5.0599	3.6
11	17.7828	1.2	37	25.9002	1.8	63	5.0757	3
12	4.9480	2.4	38	25.9002	1.8	64	5.0599	3.6
13	10	3	39	5.0913	2.4	65	5.2299	1.2
14	4.9582	3	40	4.8765	2.4	66	4.8854	1.8
15	4.9480	2.4	41	5.1275	3	67	9.5992	1.8
16	4.8986	1.2	42	4.9248	2.4	68	5.0757	3
17	7.8418	1.8	43	13.5120	3.6	69	5.2299	1.2
18	23.4035	1.2	44	5.1275	3	70	5	2.4
19	4.8986	1.2	45	14.5648	1.8	71	4.9248	2.4
20	7.8418	1.8	46	9.5992	1.8	72	5	2.4
21	5.0502	6	47	5.0458	3	73	5.0785	1.2
22	5.0458	3	48	4.8549	2.4	74	5.0913	2.4
23	10	3	49	5.2299	1.2	75	4.8854	1.8
24	5.0502	6	50	19.6799	1.8	76	5.0785	1.2
25	13.9158	1.8	51	4.9935	4.8			
26	13.9158	1.8	52	5.2299	1.2			

5. Experimental Results

In this section we have examined basic Benders decomposition algorithm and improved algorithm on a real world network with 24 nodes and 76 links. Our test network which is widely used in traffic assignment literature, is the network of Sioux-Falls city streets. This network is shown in Figure 2. In this figure, the numbers beside each link denotes the ID of corresponding link. We have used this network because finding the most vital links of network over time has some application on traffic management. Table 1 represents the capacity and traverse time of all links of Sioux-Falls network. To find single most vital link over time we used both basic and improved algorithms. We have also compared both basic and improved algorithms with branch and bound method on a large scale random grid (40× 40) network with 1600 nodes and 3120 links. We have written the code of both basic and improved algorithms in MATLAB and for the branch and bound method we have used LINGO. These numerical experiments have been conducted with double precision arithmetic on a Laptop computer with Intel(R) Core(TM)2Duo CPU 2.00 GHz and 3 GB RAM, using the Microsoft Window 7 operating system.

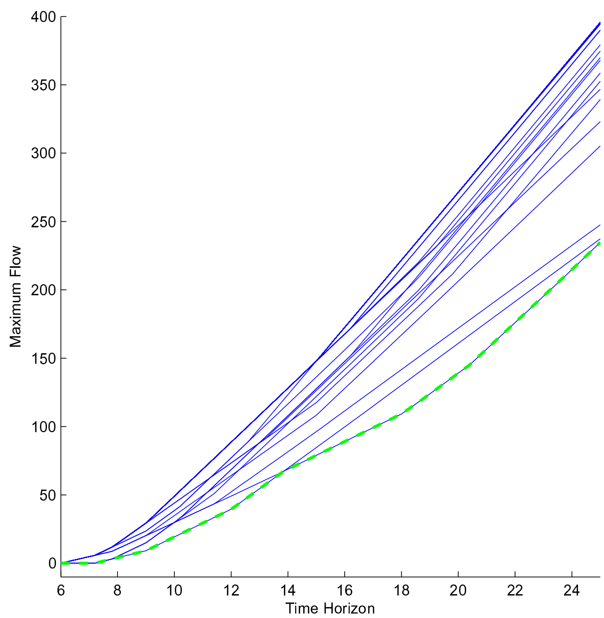

Figure 3. Functions $M_a(T)$ and $M_{a_v(T)}(T)$ for Sioux-Falls network

In Table 2 we have found single most vital link, 2 most vital links and 3 most vital links over time for (s,t) pair (16,22). First column of Table 2 shows time horizons for which we have found most vital links. Besides first column, Table 2 has 3 other columns corresponding to $k = 1, 2$ and 3. In each of these columns first sub-column shows k most vital links with respect to time horizon T, and second sub-column shows maximum flow that can be sent from node 16 to node 22 up to time T through the network in which k most vital links are blocked.

Table 2 confirms the result of Theorem 4. Note that, according to Table 2, the single most vital link (i.e. (16,18)) remains unchanged for all time horizons greater than 85. In Section 4 it is mentioned that a result same as Theorem 4 holds for k-most vital links over time problem. Table 2 confirms this claim; that is, for all time horizons greater than 85, two most vital links and for time horizons greater than 155, three most vital links remain unchanged.

In Figure 3 we have plotted all 76 functions $\{M_a(T)|a \in A\}$ for Sioux-Falls network. However, for this network most of functions $M_a(T)$ coincide with each other. In this figure we have plotted the function $M_{a_v(T)}(T)$ with a green dashed line. Notice that as Theorem 3 claims, the function $M_{a_v(T)}(T)$ is an increasing, piecewise linear, continuous and quasi-convex function.

Our experiments showed that to plot function $M_{a_v(T)}(T)$ as minimum of functions $M_a(T)$ within time interval [6,25] takes 187.5723 seconds. A same experiment showed that to plot function $M_{a_v(T)}(T)$ by applying branch and bound method on (11) takes 10.3245 seconds while to plot function $M_{a_v(T)}(T)$ using our improved algorithm takes just 4.77 seconds. In each of these experiments we selected step size 0.2 to plot functions in MATLAB and LINGO.

Notice that for time horizon $T = 150$ single most vital link is $(16, 18)$ but the set of 3 most vital links for the same time horizon (i.e. $\{(15, 22), (20, 22), (21, 22)\}$) does not contain the link $(16, 18)$. This observation demonstrates that for fixed time horizon T, if a link belongs to the set of k most vital links, it does not necessarily belong to the set of $k + l$ most vital links over time for $l = 1, 2, \cdots$.

Our experiments confirmed the Corollary 1. We fond that the interval $(0, +\infty)$ can be partitioned into time intervals (0,7.2),(7.2, 7.8),(7.8,13.8533), (13.8533,25.55) and (25.55,$+\infty$) such that $A_v(T)$ is constant for all T in each of these time intervals.

In Table 3 we have compared the basic Benders decomposition algorithm, improved algorithm and Branch and Bound (B&B) method with each other. We obtained the results presented in Table 3 by applying the branch and bound method and both of proposed algorithms in this paper on a grid 40×40 random network with 1600 nodes and 3120 links for (s,t) pair (1,1600). In this grid network the shortest path length between node 1 and node 1600 is 419.2826. Therefore for given time horizons greater than 419.2826 a positive flow arrives to node 1600 up to given time horizon. In Table 3 we examined four-time horizons 420, 455, 480 and 500. For each of these time horizons we applied branch and bound method, basic Benders decomposition algorithm and improved Benders decomposition algorithm on grid network. For each algorithm we have recorded the algorithm running time and number of iterations that each algorithm performs to find SMVLOT of network for corresponding time horizon. We have also recorded $M_{a_v(T)}(T)$ for each algorithm to recognize possible deviation of algorithms. As is clear from Table 3, all of these methods converges without deviation from optimal solution.

Table 3 demonstrates the efficiency of the both basic and improved Benders decomposition algorithms versus branch and bound method. Specially, the improved Benders decomposition algorithm is rapidly convergent due to its fully combinatorial nature that solves no binary programming problem analytically. The number of iterations in both of Benders decomposition algorithms are meaningful, which declares that in these algorithms optimal solution achieved generating only a small fraction of extreme points (i.e. $\hat{X}$). As is shown in Table 3, our experiments showed that for time horizons near to the length of shortest path branch and bound method is very time consuming. For example when $T = 420$, the branch and bound method finds the SMVLOT in grid network , after 27.56 minutes and performing 4126291 iterations. Table 3 demonstrates the superiority of basic and improved Benders decomposition algorithms in contrast with branch and bound method. When compared with improved Benders decomposition algorithm, basic Benders decomposition algorithm suffers from slow convergence. This is reasonable since in each iteration the basic Benders decomposition algorithm solves a binary linear programming by analytical methods(we have used branch and bound method).

6. Conclusion

In this paper we have introduced single and k-most vital links over time problems and we have formulated these problems as a mixed binary linear programming problem. Due to special structure of

Table 2. Most vital links over time for (s,t) pair (16,22).

	k=1		k=2		k=3	
time horizon	$a_v(T)$	max flow	vital links	max flow	vital links	max flow
$T = 7.3$	(15,22)	0.5076	$(15,22)$ $(20,22)$	0		
$T = 9.5$	(15,22)	14.20	$(15,22)$ $(20,22)$	2.53	$(15,22)$ $(20,22)$ $(16,18)$	0
$T = 14$	(18,20)	68.87	$(16,17)$ $(18,20)$	30.10	$(16,10)$ $(16,17)$ $(18,20)$	0
$T = 26$	(15,22)	252.69	$(15,22)$ $(20,22)$	157.26	$(15,22)$ $(20,22)$ $(21,22)$	69.85
$T = 85$	(16,18)	1155.44	$(16,17)$ $(16,18)$	746.17	$(15,22)$ $(20,22)$ $(21,22)$	364.85
$T = 150$	(16,18)	2138.94	$(16,17)$ $(16,18)$	1389.73	$(15,22)$ $(20,22)$ $(21,22)$	689.85
$T = 155$	(16,18)	2214.59	$(16,17)$ $(16,18)$	1439.22	$(16,8)$ $(16,17)$ $(16,18)$	714.64
$T = 200$	(16,18)	2895.47	$(16,17)$ $(16,18)$	1884.76	$(16,8)$ $(16,17)$ $(16,18)$	933.11
$T = 500$	(16,18)	7434.67	$(16,17)$ $(16,18)$	4854.98	$(16,8)$ $(16,17)$ $(16,18)$	2389.59
$T = 10^3$	(16,18)	15000	$(16,17)$ $(16,18)$	9805.35	$(16,8)$ $(16,17)$ $(16,18)$	4817.05

Table 3. Comparison of basic Benders decomposition algorithm, improved algorithm and Branch and Bound(B&B) method on a 40× 40 grid network.

	$T = 420$			$T = 455$		
	run time(min.)	$M_{a_v(T)}(T)$	itrerations	run time(min.)	$M_{a_v(T)}(T)$	itrerations
B&B	27.56	3.3525	4126291	10.03	207.45	1507383
Basic Alg.	19.62	3.3525	162	6.81	207.45	43
Improved Alg.	3.34	3.3525	183	1.23	207.45	46
	$T = 480$			$T = 500$		
	run time(min.)	$M_{a_v(T)}(T)$	itrerations	run time(min.)	$M_{a_v(T)}(T)$	itrerations
B&B	7.37	495.74	1353420	9.17	623.93	1521196
Basic Alg.	5.23	495.74	48	6.81	623.93	57
Improved Alg.	1.34	495.74	50	1.73	623.93	68

single and k-most vital links over time problems, we have proposed a basic Benders decomposition algorithm for solving them. We have adopted an iterative method to improve basic Benders decomposition algorithm for determining single most vital link of network over time. The improved algorithm solves no integer programming problem directly, therefore has a fully combinatorial nature. We have also demonstrated that starting from zero the single and k-most vital links change over time but there exists a time horizon $\tilde{T}$ such that most vital links remain unchanged for time horizons greater than $\tilde{T}$. We have proved that the function of maximum flow over time between a pair of nodes through a network in which k most vital links over time are blocked, is an increasing, piecewise linear, continuous and quasi-convex function with respect to time horizon. Finally we have examined our proposed algorithms and mixed binary linear programming model on a small size and a large scale network.

Since introduction of over time flows, this issue has provided wide and extensive research area for researchers. Due to their complexity, this kind of flows has not been studied widely when compared with static network flows. Therefore there exists a wide research directions for future research in this issue. We are pursuing issues of quickest flows over time and finding its most vital links in an ongoing work. As a research direction one can develop algorithms for finding the critical time horizons and specially greatest critical time horizon. The model studied in this paper may be generalized to a case that permits the flow to be stored at some nodes which is a challenging issue in network flows over time. As another potential for future research this model can be extended to a special category of flows over time in which network parameters are a function of corresponding link flows (i.e. flow dependent traverse times and capacities). This work does not consider uncertainty in determining the most vital links of the network. Pursing uncertainty issue in finding most vital links over time problem and many other research directions (e.g. determining KMVLOT in a network carrying multi-commodity flows over time and earliest arrival flows network, developing rapid algorithms) remain potentials for future research.

Acknowledgements

The authors gratefully acknowledge anonymous referees for their constructive comments that improved the presentation of this paper.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[2] D. S. Altner, zlem Ergun, and N. A. Uhan. The maximum flow network interdiction problem: Valid inequalities, integrality gaps, and approximability. *Operations Research Letters*, 38:33–38, 2010.

[3] E. Anderson, P. Nash, and A. Philpott. A class of continuous network flow problems. *Mathematics of Operations Research*, 7:501–514, 1982.

[4] E. J. Anderson and A. B. Philpott. Optimisation of flows in networks over time. In F. P. Kelly, editor, *Probability, Statistics and Optimisation*, chapter 27, pages 369–382. Wiley, New York, 1994.

[5] A. Bar-Noy, S. Khuller, and B. Schieber. The complexity of finding most vital arcs and nodes. Technical Report CS-TR-3539, University of Maryland, Institute of Advanced Computer Studies, College Park, MD, 1995.

[6] J. Benders. Partioning procedures for solving mixed integer variables programming problems. *Numerische Mathamatik*, 4:238252., 1962.

[7] R. E. Burkard, K. Dlaska, and B. Klinz. The quickest flow problem. *ZOR Methods and Models of Operations Research*, 37(1):3158, 1993.

[8] H. Corley and D. Sha. most vital links and nodes in weighted networks. *Operations Research Letters*, 1 (4):157–160, 1982.

[9] L. Fleischer and . Tardos. Efficient continuous-time dynamic network flow algorithms. *Operations Research Letters*, 23(3-5):71 – 80, 1998.

[10] L. Ford and D. Fulkerson. Constructing maximal dynamic flows from static flows. *Operations Research*, 6:419–433, 1958.

[11] B. Hoppe and . Tardos. The quickest transshipment problem. *Mathematics of Operations Research*, 25(1):3662, 2000.

[12] K.-C. Lin and M.-S. Chern. The fuzzy shortest path problem and its most vital arcs. *Fuzzy Sets and Systems*, 58:343–353, 1993.

[13] K. Malik, A. Mittal, and S. Gupta. The k most vital arcs in the shortest path problem. *Operations Research Letters*, 8:223–227, 1989.

[14] K. G. Murty. *Linear Programming*. John Wiley and Sons, New York, 1983.

[15] A. Orda and R. Rom. On continuous network flows. *Operations Research Letters*, 17:27–36, 1995.

[16] M. Skutella. An introduction to network flows over time. In W. Cook, L. Lovasz, and J. Vygen, editors, *Research Trends in Combinatorial Optimization*. Springer, Berlin, 2009.

[17] R. D. Wollmer. Some methods for determining the most vital link in a railway network. *RAND Memorandum RM-3321-ISA*, 1963.

[18] R. K. Wood. Deterministic network interdiction. *Mathematical and Computer Modelling*, 17:1–18, 1993.

Shahram Morowati-Shalilvand (Master of Science) received his M.Sc. degree in operations research from University of Tabriz in 2010. His research interest includes network flows, network interdiction and traffic assignment.

Javad Mehri-Tekmeh (Assistant Professor) received his Ph.d. degree in operations research from University of Tabriz in 2006. Since 2006 he has been at University of Tabriz. His research interest includes operations research, computer science and graph theory.