Adv Syst Sci Appl 2019; **01**; 1-11
Published online at https://ijassa.ipu.ru/index.php/ijassa/article/view/656

A simple improved algorithm to find all the lower boundary points in a multiple-node-pair multistate flow network

Majid Forghani-elahabad[1*], Nelson Kagan[2]

[1)] *Centro de Matemática, Computação e Cognição – CMCC, Universidade Federal do ABC – UFABC, Santo André, SP, Brazil*

E-mail: majid@usp.br

[2)] *Department of Electrical Engineering, University of Sao Paulo, Sao Paulo, Brazil*

E-mail: nelsonk@pea.usp.br

Received October 17, 2018; Revised February 20, 2019; Published April 10, 2019

Abstract: There are several real-world systems which can be modeled as multistate flow networks (MFNs). Many researchers have worked on reliability evaluation of an MFN, and accordingly variety of approaches have been proposed in this regard. The problem of finding all the lower boundary points (LBPs) in a multistate flow network (MFN) has been attracting significant attention in the recent decades as these vectors can be used for reliability evaluation of MFNs. Once all the LBPs are determined, the reliability of the network can be calculated by using some techniques such as the inclusion–exclusion principle or the sum of disjoint products (SDP). Several algorithms have been proposed to address this problem in the literature in terms of minimal paths or minimal cuts. As in the real systems the data (flow or commodity) are usually transmitting from several sources to a number of destinations, the problem of determining all the LBPs in a multi-node-pair MFN is considered in this work. An improved algorithm is proposed to address the problem. The algorithm is illustrated through a benchmark network example. The complexity results are computed. The efficiency of the algorithm is demonstrated through a numerical example and the complexity results.

Keywords: Lower boundary points; Multi-node-pair multistate flow network; minimal paths; *d-MP* problem; System reliability; Sum of disjoint product.

1. INTRODUCTION

Due to ever-increasing importance of transport networks such as communication or telecommunication networks, there is certainly need for designing and constructing the most possible reliable systems. A transport network can be generally modeled as a multistate flow network (MFN, also called stochastic-flow network) whose arcs (and possibly nodes) have randomly variable capacities due to failures, the need for maintenance, operations activities, etc. [1-22]. Traditionally, several indices have been proposed and employed to measure the reliability of an MFN in the literature. The most attractive reliability index is the two-terminal reliability (TTR) or the source-destination reliability index [2-4, 14]. For a given demand value of d, the TTR in an MFN is the probability of transmitting at least d units of flow (data or commodity) from a source node to a destination node [5-9]. However, the realistic networks such as communication or telecommunication networks need to transmit concurrently several demands of signals (flow or data) from various source nodes to a number of destination nodes, and consequently the problem of reliability evaluation of multi-node-pair MFN (MNP-MFN) has been emerged [5, 8, 14, 16]. In an MNP-MFN, there is a

[*] Corresponding author: Email address: majid@usp.br, Phone: +55(11)3091-5317, Fax: +55(11)3091-5404.

set of node pairs with related demand values which should be satisfied. Therefore, the reliability of such a system is defined as the probability of transmitting all the required demands from the sources to the corresponding destinations concurrently [5, 8, 14, 16]. Although many algorithms have been proposed in the literature to address the problems, evaluating the system reliability is an NP-hard problem [3], and thus the problem continues to be interesting to investigate.

To evaluate the reliability of an MFN with single source and destination nodes, many algorithms have been proposed in terms of minimal cuts (upper boundary points – UBPs or *d*-MCs) [5-8, 12, 15, 21] or minimal paths (lower boundary points – LBPs or *d-MP*s) [9-11, 13, 14, 16]. Once all the UBPs or LBPs are determined, the reliability can be calculated by using some techniques such as the inclusion–exclusion principle [6] or the sum of disjoint products (SDP) [2]. In [13], the notion of *d-MP* candidate was introduced, and it was demonstrated that the minimal elements of all the *d-MP* candidates are the set of all the LBPs. Afterwards, several algorithms have been proposed in the literature to address the problem [9-11, 13, 14]. The algorithms generally consist of two main stages; finding all the *d-MP* candidates, and then checking each candidate for being a *d-MP*.

The focus of this work is on determining all the *d-MP*s (or LBPs) in an MNP-MFN. In general, to compute the system reliability of an MNP-MFN, one can find all the system state vectors under which all the given demands can be satisfied simultaneously, and then calculate the probability of such vectors according to the given probability distributions of the arcs' capacities. To search for such vectors, it is usually assumed that all the required minimal paths or minimal cuts are at hand in advance [5-11, 13, 14], and after determining these vectors, the system reliability can be computed using some techniques such as the inclusion-exclusion principle [6] or the sum of disjoint products [2]. Lin and Yuan [14] considered an MNP-MFN and proposed an algorithm to evaluate the system reliability in terms of minimal paths. The algorithm first finds all the flow patterns which satisfy the demands, then transforms these patterns to system-state vectors (SSVs) which are *D-MP* candidates, and finally checks each candidate for being a minimal one to find all the LBPs. It is reminded that as there is a set D of the demand values in the multi-pair-node case, the SSVs are called *D-MP* rather than *d-MP*. The most time consuming part of the proposed algorithm in [14] is the checking process as it needs to compare all the candidates to each other. In [16], Lin and Yeng applied this algorithm to a real-world network, namely Taiwan Advanced Research and Education Network, for evaluating the reliability of a computer network with multiple sources. Chakraborty and Goyal [5] proposed a three-step algorithm in terms of minimal cuts which uses SDP technique to evaluate the reliability of an MNP-MFN. The notable point on the algorithm is that it considers all the arcs with two states of failure or fully working, whereas usually in the real-world networks it is possible to have multistate arcs.

Here, considering the problem of determining all the LBPs in an MNP-MFN, some new results are presented. Based on the presented results, a simple improved algorithm is proposed to address the problem. The algorithm is illustrated through a benchmark network. The complexity results are computed, and accordingly the proposed algorithm is shown to be more efficient than the existing ones in the literature. The reminder of this work is organized as follows. Section 2 states the required notations and explains some preliminaries on computing the system reliability. In Section 3, an algorithm is proposed and illustrated through a benchmark example. Then, the complexity results are computed, and the algorithm is compared with an existing one in the literature. Section 4 concludes the work.

2. PRELIMINARIES

Here, first the required notations, nomenclature, and assumptions are stated, and then some preliminaries on computing the system reliability are described.

2.1. Notations, Nomenclature, and Assumptions

Here, the required notations and assumptions are stated.

Table 2.1. The required notations.

$G(N, A, M, Q)$	An MNP-MFN with $N = \{1, 2, ..., n\}$ being the set of nodes (n is the number of nodes), $A= \{a_i \mid 1 \le i \le m\}$ being the set of arcs (m is the number of arcs), $M = (M^1, M^2, ..., M^m)$ being a vector, where M^i denotes the maximum capacity of arc a_i, for $i = 1, 2, ..., m$, and $Q = \{(i,j) \mid i$ is a source node with corresponding destination node $j\}$ being the set of all the node pairs.
x_i	x_i denotes the current capacity of the arc a_i, for $i = 1, 2, \ldots, m$.
X	$X= (x_1, x_2, ..., x_m)$ is a system-state vector representing the current capacities of all the arcs.
$d_{i,j}$	is a given demand of data (flow) to be transmitted from source node i to destination node j for every node pair $(i, j) \in Q$.
v, w	are the number of sources and destinations in the network, respectively.
o_i	$o_i = \sum_{j=1}^{w} d_{i,j}$ is the total flow outgoing from source node i, for $i = 1, \ldots, v$.
u_j	$u_j = \sum_{j=1}^{v} d_{i,j}$ is the total flow incoming to destination node j, for $j = 1, \ldots, w$.
s, t	are the added artificial source and destination nodes to the network.
b_i	is the added artificial arc from node s to the source node i with the fixed capacity of o_i, for $i = 1, \ldots, v$.
c_j	is the added artificial arc from the destination node j to node t with the constant capacity of u_j, for $j = 1, \ldots, w$.
$\bar{G}$	is the new network after adding artificial nodes and arcs into G.
V(X)	is the maximum flow on the network $\bar{G}$ from node s to node t.
Z(X)	$= \{a_i \in A \mid x_i > 0\}$ is the set of nonzero-capacity arcs, under X.
e_i	$= 0(a_i)$ is a system-state vector in which the capacity level is 1 for a_i and 0 for the other arcs.
$H^{i,j}$	is the set of all the minimal paths from source node i to destination node j, for every $(i,j) \in Q$.
H	is the set of all the minimal paths in the network G, i.e., $H = \bigcup_{(i,j)\in Q} H^{i,j}$.
h	is the number of all the minimal paths in G, i.e., $h = \lvert H \rvert$.
P_j	is the jth minimal path in G, for $j = 1, \ldots, h$.
CP_j	$= min\ \{M^i \mid a_i \in P_j\}$ is the capacity of P_j, for $j = 1, 2, \ldots, h$.
D	$= \{d_{i,j} \mid (i, j) \in Q\}$ is the set of all the demands in the network.
σ	is the number of all the *D-MP* candidates.

A system state $X = (x_1, x_2, ..., x_m)$ is less than or equal to system state $Y = (y_1, y_2, ..., y_m)$, written as $X \leqslant Y$, when $x_i \leqslant y_i$, for $i = 1, 2, ..., m$, and also $X < Y$, when $X \leqslant Y$ with at least one j, $j = 1, 2, ..., m$, such that $x_j < y_j$. A vector $X \in \Psi$ is said to be a minimal vector when there is no vector $Y \in \Psi$ such that $Y < X$. A path is a sequence of adjacent arcs from a source node to a destination node, and a minimal path (MP) is a path with no cycle. The following assumptions are considered in this work:

1. Each node is perfectly reliable.
2. Each added artificial arc in $\bar{G}$ is perfectly reliable.
3. Flow in G and $\bar{G}$ satisfies the flow conservation law [1].
4. The capacity of each arc a_i in A is modelled as a non-negative integer-valued random number less than or equal to M_i.
5. The arcs' capacities are statistically independent.

2.2. Reliability

In a multi-node-pair multistate flow network (MNP-MFN), for each node pair (i, j) in Q, there is a demand d_{ij} of flow (data or commodity) to transmit from source node i to destination node j. Hence, the system reliability is the probability of all the demands being satisfied, that is for each node pair (i, j) in Q, at least d_{ij} units of flow (data) can be transmitted from node i to node j, simultaneously. To compute such a probability, one can first find the set of all the system-state vectors (SSVs) on which all the demands can be satisfied simultaneously, and then compute the probability of those SSVs. Thus, considering Ψ = {X | X is an SSV on which all the demands can be satisfied concurrently}, we see that $R_D = \Pr\{X \mid X \in \Psi\}$. Now, assuming $\Psi_{\min} = \{X \mid X$ is a minimal vector in $\Psi\} = \{X^1, X^2, \ldots, X^\delta\}$ and $A^i = \{X \mid X \geq X^i\}$, for $i = 1, 2, \ldots, \delta$, it is seen that $\Psi = \bigcup_{i=1}^{\delta} A^i$, and consequently the system reliability can be computed using methods such as inclusion-exclusion technique or sum of disjoint products (SDP) [2, 5]. For example, in SDP technique, it is set $B^1 = A^1$ and $B^r = A^r - \bigcup_{i=1}^{r-1} A^i$ for each $r = 2, \ldots, \delta$, and then

$$R_D = \sum_{i=1}^{\delta} \Pr(B^i) \tag{2.1}$$

Where $\Pr(B^i) = \sum_{X \in B^i} \Pr(X)$, and $\Pr(X) = \prod_{1 \leq j \leq m} \Pr(x_j)$. Hence, to evaluate the reliability of an MNP-MFN, it is enough to find all the set $\Psi_{\min}$ which is the set of all the *lower boundary points* (LBPs) in the network. The problem of finding all the LBPs is called *D-MP* problem in the literature [8, 13, 14, 16].

3. MAIN BLOCK

To find all the LBPs in an MNP-MFN, we first define the notion of flow pattern at level D (called D-FP) as follows. It is reminded that D is the set of all the demand values.

Definition 3.1:
A vector $F = (f_1, f_2, \ldots, f_h)$ is named a flow pattern at level D (D-FP) when it satisfies the followings:

$$\begin{cases} (I) \sum_{l: P_l \in H^{i,j}} f_l = d_{i,j} \quad , \quad \text{for all } (i, j) \in Q, \\ (II) \sum_{l: a_i \in P_l} f_l \leq M_i \quad , \quad i = 1, 2, \ldots, m. \end{cases} \tag{3.1}$$

It is noted that in a *D-FP*, the component f_l shows the amount of flow on minimal path (MP), P_l, and this is why it is called a flow pattern. In the system (3.1), the first equation guarantees that the summation of flows on all the MPs from node i to node j is $d_{i,j}$, and so the given

demand can be satisfied. The second inequality guarantees that the summation of flows on each arc is less than the maximum capacity of the arc.
For example, considering $Q = \{(1, 2), (1, 3), (4, 3)\}$ as the set of node pairs in Fig. 1, the set of MPs are $H^{1,2} = \{\{a_1\}, \{a_5, a_4\}\}$, $H^{1,3} = \{\{a_1, a_2\}, \{a_5, a_6\}, \{a_1, a_3, a_6\}, \{a_5, a_4, a_2\}\}$, $H^{4,3} = \{\{a_6\}, \{a_4, a_2\}\}$, and hence $H = \{\{a_1\}, \{a_5, a_4\}, \{a_1, a_2\}, \{a_5, a_6\}, \{a_1, a_3, a_6\}, \{a_5, a_4, a_2\}, \{a_6\}, \{a_4, a_2\}\}$. Now, considering $d_{1,2} = 1$, $d_{1,3} = 3$, and $d_{4,3} = 2$ as the given demand values, and $M = (3, 3, 2, 3, 2, 2)$ as the max-capacity vector, the set of all the *D-FP*s is obtained as follows. (0,1,3,0,0,0,2,0), (1,0,2,0,0,1,2,0), (0,1,2,0,0,1,2,0), (1,0,1,0,0,2,2,0), (0,1,2,0,1,0,1,1), (1,0,2,1,0,0,1,1), (0,1,2,1,0,0,1,1), (1,0,1,0,1,1,1,1), (0,1,1,0,1,1,1,1), (1,0,1,1,0,1,1,1), (1,0,0,0,1,2,1,1), (0,1,1,0,2,0,0,2), (1,0,1,1,1,0,0,2), (0,1,1,1,1,0,0,2), (1,0,1,2,0,0,0,2), (1,0,0,0,2,1,0,2), (1,0,0,1,1,1,0,2).

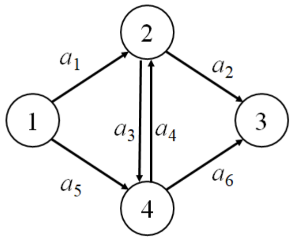

Fig. 1. A benchmark example

It is reminded that each *D-FP* shows the amount of flow on all the MPs, and hence for each *D-FP*, say $F = (f_1, f_2, \ldots, f_h)$, one can find the corresponding system-state vector (SSV), $X = (x_1, x_2, \ldots, x_m)$ using the following equation.

$$x_i = \sum_{l:a_i \in P_l} f_l\,,\ i = 1, 2, \ldots, m. \tag{3.2}$$

The SSV, X which is calculated through Eq. (3.2) above is called a *D-MP* candidate. A notable point is that as the number of MPs is usually far greater than the number of arcs in a network, it is possible to obtain duplicate *D-MP* candidates from different *D-FP*s, and so it is required to remove the duplicate candidates. For example, transforming the above-mentioned seventeen *D-FP* vectors in Fig. 1, only five different SSVs $X^1 = (3, 3, 0, 1, 1, 2)$, $X^2 = (2, 3, 0, 2, 2, 2)$, $X^3 = (3, 3, 1, 2, 1, 2)$, $X^4 = (2, 3, 1, 3, 2, 2)$, $X^5 = (3, 3, 2, 3, 1, 2)$ are obtained and the other twelve SSVs are duplicates. The following result proven in [13, 14] shows that to find all the LBPs, it is enough to first find all the *D-MP* candidates and then check each candidate for being an LBP.

Lemma 3.1:
Each LBP is a D-MP candidate.

Proposition 3.1:
There is a possibility of generating duplicate D-MP candidates from different D-FPs through Eq. (3.2).

In fact, it can be considered as a common drawback in the presented works in the literature to not notice for generating duplicate candidates from different flow pattern vectors [8]. The first trivial method is to compare *D-MP* candidates for detecting and removing the duplicates. However, as they are *m*-tuple vectors, the time complexity of comparison process is relatively high. Hence, the proposed data structure in [7] is used here instead of comparing the vectors. Next, it is briefly explained how the duplicate candidates can be removed using the proposed data structure in [7]. Let $M^* = max\ \{M_1, M_2, \ldots, M_m\}$, $X = (x_1, \ldots, x_m)$ be a *D-*

MP candidate, k be the number of digits in M^*, and k_i be the number of digits in x_i, for i = 1, 2, …, m. Since $x_i \le M_i \le M^*$, for i = 1, …, m, it is clear that $k_i \le k$. Now, to construct a k-digit number n_i for component x_i, if $k = k_i$, then let $n_i = x_i$, and if $k_i < k$, then let $n_i = \overline{\underbrace{00...0}_{k-k_i \text{ times}} x_i}$. Thereby, the ml-digit number $N_X = \overline{n_1 n_2 ... n_m}$ can be constructed associated with X.

Corollary 3.1:

The time complexity of removing the duplicate D-MP candidates by using the method above is O(σlogσ), where σ is the number of all the candidates.

It should be noted that under each *D-MP* candidate X obtained by Eq. (3.2), when the flows are transmitting simultaneously from all the sources to all the destinations, the maximum flow from each source node i to the corresponding destination node j equals the sum of flows on all the MPs between them, that is exactly $\sum_{l:P_l \in H^{i,j}} f_l = d_{i,j}$. Hence, the following result is at hand.

Proposition 3.2:

Under D-MP candidate X, the maximum flow between each node pair (i,j) *in Q is exactly* $d_{i,j}$ *when the flows are transmitting concurrently from all the sources to all the destinations.*

To find all the LBPs from all the candidates, the authors in [13, 14, 16] compared all the candidates to detect the minimal ones. However, the number of candidates grows exponentially with the size of network and as they are m-tuple vectors, it is really time consuming to compare all the candidates. Here, an improved technique is proposed to check each candidate to be an LBP. In fact, two artificial nodes and a few artificial arcs are added into the network to make use of max-flow algorithm for checking the *D-MP* candidates.
For a given MNP-MFN, say G, first two artificial nodes s (a source) and t (a destination) are added to G. Then, from node s to each source node i, an artificial arc b_i with the fixed capacity of $o_i = \sum_{j=1}^{w} d_{i,j}$ is added, for i = 1, 2, …, v. Moreover, from each destination node j to node t, an artificial arc c_j with the fixed capacity of $u_j = \sum_{i=1}^{v} d_{i,j}$ is added, for j = 1, 2, …, w. The new network with artificial nodes and arcs is named $\bar{G}$. For example, considering Q = {(1, 2), (1,3), (4,3)} as the set of node pairs, and D = {1, 3, 2} as the set of demand values in the given network in Fig. 1, the new network $\bar{G}$ with artificial nodes and arcs is constructed as given in Fig. 2. The capacities of b_1, b_2, c_1, and c_2 are respectively o_1 = 4, o_2 = 2, u_1 = 1, and u_2 = 5.

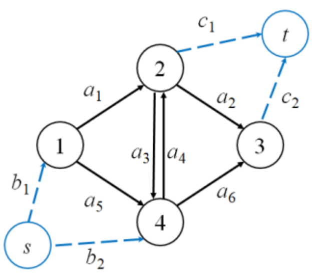

Fig. 2. The benchmark with added artificial nodes and arcs.

As the added artificial arcs are perfectly reliable with a constant capacity, they are not considered in the system state vectors (SSVs) of the network, and an SSV in $\bar{G}$ is still exhibited as X= (x_1, x_2, ..., x_m) where m is the number of arcs in G. It is reminded that the capacity of an artificial arc from node s to source node i is always assumed to be o_i, for each $i = 1, \ldots, v$, and the capacity of an artificial arc from destination node j to node t is always assumed to be u_j, for $j = 1, \ldots, w$. This way, it is possible to check all the node pairs' demands by sending $d = \sum_{i=1}^{v} o_i = \sum_{j=1}^{w} u_j$ units of flow from node s to node t.

Corollary 3.2:

$$\sum_{i=1}^{v} o_i = \sum_{j=1}^{w} u_j \ .$$

Proof. According to definition, we have $o_i = \sum_{j=1}^{w} d_{i,j}$, and consequently it is concluded that $\sum_{i=1}^{v} o_i = \sum_{i=1}^{v}\sum_{j=1}^{w} d_{i,j} = \sum_{j=1}^{w}\sum_{i=1}^{v} d_{i,j} = \sum_{j=1}^{w} u_j$. □

Lemma 3.2:

If X is a D-MP candidate, then V(X) = d in $\bar{G}$ *, namely the maximum flow from node s to node t equals d, where* $d = \sum_{i=1}^{v} o_i = \sum_{j=1}^{w} u_j$.

Proof. First, the d units of flow from node s are transmitted through the artificial arcs b_is to the source nodes (o_i units on arc b_i, for $i = 1, 2, \ldots, v$), and then according to Proposition 2, these d unites of flow are transmitted from the source nodes to the destination nodes so that u_j units of flow is received at destination node j, for $j = 1, 2, \ldots, w$. Finally, according to considered capacities for the artificial arcs, the flows are transmitted to node t through the arcs c_js. This shows that V(X) = d which completes the proof. □

Theorem 3.1:

A D-MP candidate X is an LBP if and only if for each a_i *in Z(X), we have* $V(X - e_i) < d$ *in* $\bar{G}$.

Proof. According to Lemma 1, each LBP is a *D-MP* candidate, and thus the first side has been already demonstrated. Now, assume that X is a candidate which is not an LBP (or equivalently a minimal vector in Ψ). Thus, there is a vector Y in Ψ such that X > Y. Without loss of generality, assume that $x_l > y_l$, for an l. Since Y is a vector in Ψ, all the demands are satisfied under Y, and so we have V(Y) ≥ d in $\bar{G}$. Thus, we have $V(X - e_l) \geq V(Y) \geq d$, and this contradicts the definition of X being a candidate. Hence, the proof is complete. □

Considering σ as the number of all the *D-MP* candidates, since each candidate is an m-tuple vector, the time complexity of comparing them is $O(m\sigma^2)$. Whereas considering the time complexity of max-flow algorithm as $O(mn)$ [21], and that in the worst case the condition $V(X - e_i) < d$ should be checked for all the m arcs, the time complexity of checking each candidate is $O(m(m+v+w)(n+2))$. Therefore, the time complexity of checking all the candidates is $O(mn\sigma(m+v+w)) \approx O(m^2n\sigma)$ as the number of all the sources and sinks are usually less than m. It is clearly seen that the number of all the *D-MP* candidates, σ, is really far greater than mn, and thus our technique proposed here is highly more efficient than the proposed ones in [14, 16].

Lemma 3.3:

The time complexity of using Theorem 2 to check each D-MP candidate for being an LBP is $O(m^2n\sigma)$.

Now, we are at the position of stating our proposed algorithm for determining all the LBPs in an MNP-MFN.

3.1. The Proposed Algorithm

Algorithm 1. (Find all the LBPs in an MNP-MFN)

Input: An MNP-MFN, $G(N, A, M, Q)$ along with the set of all the demand values, D.

Output: The set of all the LBPs.

Step 1. Let $\Gamma = \phi$ (set of all the LBPs) and $\Lambda = \phi$ (set of all the associated numbers). Then, compute $o_i = \sum_{j=1}^{w} d_{i,j}$, for $i = 1, \ldots, v$, $u_j = \sum_{i=1}^{v} d_{i,j}$, for $j = 1, \ldots, w$, and $d = \sum_{i=1}^{v} o_i$. Afterwards, add the artificial nodes s and t and the related artificial arcs with fixed capacities into G, as explained in preceding section for constructing $\bar{G}$.

Step 2. Find a *D-FP*, say F = (f_1, f_2, …, f_h) by solving the system (3.1). If there is no solution, skip to Step 7.

Step 3. Considering *D-FP*, *F*, compute the SSV, $X_F = (x_1, x_2, ..., x_m)$ through Eq. (3.2).

Step 4. Compute the associated number N_X with X_F. If it is duplicate number, return to Step 2 to find the next *D-FP*, else add N_X to Λ and determine $Z(X_F) = \{a_i \text{ in } A \mid x_i > 0\}$.

Step 5. If $Z(X_F) = \phi$ then X_F is a LBP, add it to Γ and return to Step 2 to find the next *D-FP*.

Step 6. Select an arc in $Z(X_F)$, say a_i, and remove it from $Z(X_F)$. If $V(X_F - e_i) < d$ in $\bar{G}$, then return to Step 5, else X is not a LBP, return to Step 2 to find the next *D-FP*.

Step 7. Set Γ is the set of all the LBPs. Stop.

The following example is provided to illustrate Algorithm 1.

Example 3.1:

There are two smart meters connected to two control centers in a smart grid. Each smart meter may need to transmit some data on the network to one or both control centers. Assume that the network is the one given in Fig. 1 *in which nodes* 1 *and* 4 *are the smart meters and nodes* 2 *and* 3 *are the control centers. As it is seen in Fig. 1, there are some arcs* (*links*) *between the smart meters and control centers to transmit data on. Suppose that the first smart meter* (*node* 1) *needs to send one unit of data to the first control center* (*node* 2) *and three units of data to the second control center (node 3) per hour. Moreover, the second smart meter* (*node* 4) *needs to send only two units of data to the second control center* (*node* 3) *per hour. If the maximum capacities of the arcs in the network are given by* M = (3, 3, 2, 3, 2, 2), *the manager needs to know the reliability of the network to satisfy the demands. The probability distributions for the arcs' capacities are given in Table* 2.

Solution: To compute the reliability of the system, Algorithm 1 is employed as follows.
Step 1. Let $\Gamma = \phi$, $\Lambda = \phi$. Then, we have $o_1 = 1+3 = 4$, $o_2 = 2$, $u_1 = 1$, $u_2 = 2 + 3 = 5$, and $d = 6$. Afterwards, $\bar{G}$ is obtained as given in Fig. 2.
Step 2. The *D*-FP, F = (0,1,3,0,0,0,2,0) is obtained by solving the system (3.1).
Step 3. The corresponding SSV, X = (3, 3, 0, 1, 1, 2) is obtained through Eq. (3.2).
Step 4. The associated number with X is N_X = 330112 which is not duplicate, and so we have Λ = {330112} and Z(X) = {a_1, a_2, a_4, a_5, a_6}.
Step 5. $Z(X_F) \neq \phi$.
Step 6. Arc a_1 is selected and let Z(X) = {a_2, a_4, a_5, a_6}. Since V(2, 3, 0, 1, 1, 2) = 5<6, the transfer is made to Step 5.

⋮

Step 5. Z(X) = {}, and hence X is an LBP, letting Γ = {(3, 3, 0, 1, 1, 2)}, and transfer is made to Step 2.

⋮

Finally, Γ = {(3, 3, 0, 1, 1, 2), (2, 3, 0, 2, 2, 2)} is obtained. It is noted that among the seventeen *D-FP*s, only five unduplicated *D-MP* candidates are obtained containing only two LBPs. It is also noted that all the *D-FP*s and *D-MP* candidates for this example have already stated in the previous sections.

Now, to compute the system reliability by using the SDP technique, let $A^1 = \{X \mid X \geq X^1 = (3, 3, 0, 1, 1, 2)\}$, $A^2 = \{X \mid X \geq X^2 = (2, 3, 0, 2, 2, 2)\}$, $B^1 = A^1$, and $B^2 = A^2 - A^1$. Thereby, the system reliability is equal to

$$\begin{aligned}
&R_D = \Pr(B^1) + \Pr(B^2) \\
&= \Pr(x_1 \geq 3) \times \Pr(x_2 \geq 3) \times \Pr(x_3 \geq 0) \times \Pr(x_4 \geq 1) \times \Pr(x_5 \geq 1) \times \Pr(x_6 \geq 2) \\
&+ \Pr(x_1 = 2) \times \Pr(x_2 \geq 3) \times \Pr(x_3 \geq 0) \times \Pr(x_4 \geq 2) \times \Pr(x_5 \geq 2) \times \Pr(x_6 \geq 2) \\
&= 8 \times 75 \times 1 \times 95 \times 95 \times 85 \times 10^{-9} + 1 \times 75 \times 1 \times 9 \times 9 \times 85 \times 10^{-7} \\
&= 0.460275 + 0.0516375 = 0.5119125.
\end{aligned}$$

Table 1. Probability distributions for arcs' capacities of Fig. 1.

Arc	Capacity and probability			
	0	1	2	3
a_1	0.05	0.05	0.10	0.80
a_2	0.05	0.10	0.10	0.75
a_3	0.05	0.05	0.90	0
a_4	0.05	0.05	0.05	0.85
a_5	0.05	0.05	0.90	0
a_6	0.10	0.05	0.85	0

3.2. *Complexity Results*

The time complexity of Step 1 is $O(v + w)$. Assuming the number of all the *D-FP*s as σ, the complexity of solving the system (3.1) in Step 2 to find all the *D-FP*s is of order of $O(h\sigma)$ [7]. Step 3 is of order of $O(mh)$ for constructing each *D-MP* candidate, and thus is of order of $O(mh\sigma)$ for computing all the candidates. According to Corollary 1, the time complexity of Step 4 is $O(\sigma\log\sigma)$. Steps 5 and 7 are of order of $O(1)$. According to Lemma 3, the time complexity of Step 7 is $O(m^2n\sigma)$. As a result, the time complexity of Algorithm 1 is $O(mh\sigma + \sigma\log\sigma + m^2n\sigma)$. By the way, in large networks, h grows exponentially with increasing m and n, and thus $h \gg m, n$, and consequently the time complexity of Algorithm 1 is $O(mh\sigma)$ for large networks.

Theorem 3.2:
The time complexity of Algorithm 1 for large network is $O(mh\sigma)$.

3.3. *Comparison results*

Here, the proposed algorithm in [14] is compared with Algorithm 1 proposed here. As the proposed algorithm in [14] needs to solve the system (3.1) to find all the *D-FP*s, and also compares all the *D-MP* candidates to remove the duplicates and detect the minimal ones, the time complexity of this algorithm is at least $O(mh\sigma + m\sigma^2) = O(m\sigma^2)$, where the first part is for solving the system (3.1) and the second part is for the comparison process. It should be kept in mind that each *D-MP* candidate is an *m*-tuple vector. Even in small network such as the given benchmark in Fig. 1 the number of *D-FP*s, i.e., σ, is really greater than the number of MPs, i.e., h. Hence, the time complexity of the proposed algorithm in [14] is highly greater than the one for Algorithm 1, the proposed algorithm here.

It should be noted that in Example 1, the proposed algorithm in [14] needs to compare seventeen 6-tuple vectors, that is 102 comparison job, to remove the duplicates and find the five *D-MP*s, while Algorithm 1 needs only to compare seventeen numbers, that is seventeen comparison job, to do the same work. It should be noted that as the number of *D-FP*s is increased exponentially with increasing the size of the network, the practical efficiency of our proposed algorithm will be notable in large networks in comparison with the one proposed in [14].

5. CONCLUSION

Reliability evaluation of multistate flow networks (MFNs) has been a very attractive problem in the recent decades among the researchers from several research areas. Majority of the proposed algorithms have considered a single source-destination MFN whereas the real-world systems generally contain more than one node pair, and can be modeled as multiple-node-pair MFNs (MNP-MFNs). The reliability of an MNP-MFN can be computed in terms of minimal paths by determining all the lower boundary points (LBPs) or in terms of minimal cuts by computing all the upper boundary points. Here, for evaluating the reliability of an MNP-MFN, the problem of determining all the LBPs in such a network was considered. Some results were presented to improve the solution. Based on the results, a simple improved algorithm was proposed to address the problem. The algorithm was illustrated through a benchmark example. The complexity results were computed. The algorithm was shown to be more efficient than an existing one based on the complexity results and the illustrative example.

ACKNOWLEDGEMENTS

The authors thanks FAPESP (Grant 15/21924-1) for supporting this work.

REFERENCES

1. Ahuja, R.K., Magnanti, T.L. & Orlin J.B. (1993). *Network flows theory, algorithms, and applications*. New Jersey: Englewood Cliffs, Prentice-Hall Int.
2. Balan, A.O. & Traldi, L. (2003). Preprocessing minpaths for sum of disjoint products. *IEEE Transactions on Reliability*, 52 (3), 289–295.
3. Ball, M.O. (1980). Complexity of network reliability computations. *Networks* 10 (2), 153–165.
4. Boesch, F.T., Satyanarayana, A. & Suffel, C.L. (2009). A survey of some network reliability analysis and synthesis results. *Networks* 54 (2), 99–107.
5. Chakraborty, S. & Goyal, N.K. (2017). An efficient reliability evaluation approach for networks with simultaneous multiple-node-pair flow requirements. *Quality and Reliability Engineering International* 33, 1067–1082.
6. Forghani-elahabad, M. & Mahdavi-Amiri, N. (2013). On search for all d-MCs in a network flow, *Iranian Journal of Operations Research*, 4 (2), 108-126.
7. Forghani-elahabad, M. & Mahdavi-Amiri, N. (2016a). An improved algorithm for finding all upper boundary points in a stochastic-flow network. *Applied Mathematical Modelling* 40 (4), 3221–3229.

8. Forghani-elahabad, M. & Mahdavi-Amiri, N. (2016b). Reliability assessment of a multi-source multi-sink stochastic-flow network in terms of minimal paths. *Proceeding of* 9*th International Iranian Operation Research Conference*, 1065–1069.
9. Forghani-elahabad, M. & Mahdavi-Amiri, N. (2016c). A new algorithm for generating all minimal vectors for the *q* SMPs reliability problem with time and budget constraints, *IEEE Transactions on Reliability*, 65 (2), 828-842.
10. Forghani-elahabad, M. & Bonani, L. (2017). Finding all the lower boundary points in a multistate two-terminal network. *IEEE Transactions on Reliability* 66 (3), 677–688.
11. Forghani-elahabad, M. & Kagan, N. (2018). Reliability evaluation of a stochastic-flow network in terms of minimal paths with budget constraint, *IISE Transactions*, In press. DOI: https://doi.org/10.1080/24725854.2018.1504358.
12. Forghani-elahabad, M. & Kagan, N. (2019). Assessing reliability of multistate flow networks under cost constraint in terms of minimal cuts, *International Journal of Reliability, Quality and Safety Engineering*, In production.
13. Lin, J.S., Jane, C.C. & Yuan, J. (1995). On reliability evaluation of a capacitated-flow network in terms of minimal pathsets. *Networks* 25, 131–138.
14. Lin, Y.K. & Yuan, J. (2003). Flow reliability of a probabilistic capacitated-flow network in multiple node pairs case. *Computers & Industrial Engineering* 45, 417–28.
15. Lin, Y.K. & Yeh, C.T. (2011). Maximal network reliability for a stochastic power transmission network. *Reliability Engineering & System Safety* 96, 1332–39.
16. Lin, Y.K. & Yeng, L.C.L. (2013). Network reliability based decision of Internet with multiple sources and multiple sinks. *Decision Support Systems* 54, 1477–1487.
17. Locks, M.O. (1985). Recent developments in computing of system-reliability. *IEEE Transactions on Reliability* R-34 (5), 425–436.
18. Padmavathy, N. & Chaturvedi, S.K. (2013). Evaluation of mobile ad hoc network reliability using propagation-based link reliability model. *Reliability Engineering & System Safety*, 15, 1–9.
19. Ramirez-Marquez, J.E. & Rocco, C.M. (2008). All-terminal network reliability optimization via probabilistic solution discovery. *Reliability Engineering & System Safety*, 93, 1689–97.
20. Rausand, M. & Hoyland, A. (1994). *System Reliability Theory: Models, Statistical Methods, and Applications*. New York: Wiley-Interscience.
21. Orlin, J. B. (2013). Max flows in O(nm) time or better. *Proceedings of forty-fifth annual ACM Symposium on Theory of Computing*, 765–774.
22. Wu, W.W., Ning, A. & Ning, X.X. (2008). Evaluation of the reliability of transport networks based on the stochastic flow of moving objects. *Reliability Engineering & System Safety* 93, 838–44.