Hindawi
Complexity
Volume 2019, Article ID 4561845, 10 pages
https://doi.org/10.1155/2019/4561845

Research Article

Efficient Enumeration of d-Minimal Paths in Reliability Evaluation of Multistate Networks

Xiu-Zhen Xu[1,2] **Yi-Feng Niu**[1] **and Qing Li**[3]

[1]*School of Economics and Management, Chongqing University of Posts and Telecommunications, Chongqing 400065, China*
[2]*School of Computer Science and Technology, Henan Polytechnic University, Jiaozuo 454000, China*
[3]*College of Mathematics and Systems Science, Shandong University of Science and Technology, Qingdao 266590, China*

Correspondence should be addressed to Yi-Feng Niu; ifeng021@126.com

Received 15 January 2019; Revised 21 February 2019; Accepted 26 February 2019; Published 21 March 2019

Academic Editor: Dimitri Volchenkov

A number of real-world complex networks can be modeled as multistate networks for performance analysis. A multistate network consists of multistate components and possesses multiple different performance levels. For such a network, reliability is concerned with the probability of the network capacity level greater than or equal to a predetermined demand level d. One major method for multistate network reliability evaluation is using d-minimal paths. This paper proposes an efficient algorithm to find d-minimal paths. First, a new concept of qualified state vector is defined so as to fix a relatively smaller search space of d-minimal paths, and a sufficient and necessary condition for a qualified state vector to be d-minimal path is established. Then, the max-flow algorithm and the enumeration algorithm are integrated to search for d-minimal paths in the determined search space that is recursively divided into subspaces such that the searching efficiency can be increased as much as possible. Both analytical and numerical results show that the proposed algorithm is more efficient in finding all d-minimal paths. In addition, a case study related to power transmission network is performed to demonstrate the implication of network reliability.

1. Introduction

Network models have been extensively applied in analyzing real-world complex systems, such as computer and information networks [1, 2], power transmission networks [3], manufacturing networks [4], logistics networks [5–13], and gene regulatory networks [14–16]. From the viewpoint of network reliability, network model is generally classified as binary state network model and multistate network model [10]. Binary state network model assumes two states for network components: fully working or completely failed [17–20]. For example, in a gene regulatory network, the state of each gene is described by two levels: either active (fully working) or inactive (completely failed) [20], so it can be modeled as a binary state network. However, a number of actual complex systems are often affected by various random parameters due to external or internal uncertainties [21], such that they could exhibit multiple behaviors. In such situation, binary state network model is insufficient in network analysis. For example, consider a power transmission network constructed by a set of nodes (high voltage transmission towers, high voltage substations, or power plants) and a set of arcs (transmission lines). Each transmission line linking two nodes is combined with several physical lines and each physical line may provide a specific capacity or fail with a probability [3]. That is, each transmission line has several capacities due to complete failure, partial failure or maintenance, etc. In this sense, the behavior of the power transmission network is multistate. For practical applications, binary state network model has been extended and replaced by multistate network model to characterize complex behaviors of multistate networks.

A multistate network is composed of multistate components and holds a finite number of different states for performance rate. For such a network, an interesting question is to evaluate its reliability defined as the probability of the network capacity level greater than or equal to a predetermined demand level d. Researchers have presented various algorithms to evaluate the reliability of multistate networks, including direct algorithm [22–24] and indirect algorithm [25–33]. Indirect algorithm is often called

network-based algorithm established in terms of minimal paths or minimal cuts. One common indirect method for reliability computation of multistate networks is based on minimal path vectors to level d (d-minimal paths for short) [25–33]. And, all of these methods pay attention to the enumeration of d-minimal paths.

Two important types of models have been reported to solve d-minimal paths. One is proposed by Lin et al. [25], and the other is proposed by Yeh [26]. All minimal paths are assumed to be known; the model by Lin et al. [25] is constructed via the network structure and the flow conservation law. A path is a sequence of arcs that connect the source node to the sink node, and a minimal path is such a path that removing any arc will make it no longer a path. Afterward, some algorithms have been developed to find d-minimal paths in terms of the model by Lin et al. [25], such as the methods by Lin [27], Yeh [28], Chen and Lin [29], and Xu et al. [13]. It is noted that there are two drawbacks for these methods in [13, 25, 27–29]. The first is that they require all minimal paths as prior knowledge, which is an NP-hard problem, and the second is the extra burden for removing duplicate d-minimal paths.

The model by Yeh [26] requires no minimal paths knowledge, and thus possesses an advantage relative to the model by Lin et al. [25]. But, one remarkable limitation of Yeh's model is that a huge number of state vectors need to be enumerated, which leads to high computational complexity. Recently, Niu et al. [30] have attempted to improve the method of Yeh [26] by means of the concept of lower capacity bound, but the efficiency of their method highly depends on the network structure and the specified demand level d. In particular, the method by Niu et al. [30] is even inferior to the one by Yeh when the demand level is low. The method by Xu and Niu [31] is another improvement to the method by Yeh [26] and is proven to be more efficient in solving d-minimal paths.

The enumeration of d-minimal paths is a difficult task because it is an NP-hard problem [23], and developing an efficient algorithm is worthwhile from the viewpoint of reliability evaluation. Therefore, this paper focuses on the efficient enumeration of d-minimal paths in multistate network reliability evaluation. More explicitly, the incremental contributions of this paper are twofold. First, a new concept of qualified state vector is defined to determine the search space of d-minimal paths more accurately, and a sufficient and necessary condition for a qualified state vector to be d-minimal path is established. Second, the max-flow algorithm and the enumeration method are integrated to search for d-minimal paths in the determined search space that is recursively divided into subspaces so that the efficiency of finding d-minimal paths can be increased as much as possible. Both analytical and numerical results indicate that the proposed algorithm compares favorably with the existing algorithm. Finally, a case study related to power transmission network is performed to demonstrate the implications of network reliability.

The rest of this paper is arranged as follows: Section 2 introduces multistate network model and reviews the concept of network reliability. In Section 3, a concept of qualified state vector is introduced, and a sufficient and necessary condition for a qualified state vector to be d-minimal path is proposed. Moreover, the main ideas for searching for d-minimal paths are discussed, based on which an algorithm is developed along with analyses on its time complexity. In Section 4, an illustrative example and a numerical example are provided to validate the effectiveness and efficiency of the proposed algorithm. A practical case is studied in Section 5 to demonstrate the implications of network reliability. The final section presents some concluding remarks.

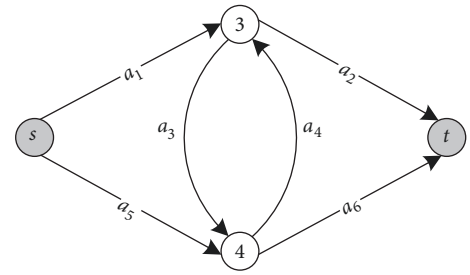

Figure 1: A multistate two-terminal network.

2. Multistate Network Model and Network Reliability

2.1. Multistate Network Model. Mathematically, a multistate network is modeled as a directed graph $G(N, A, \mathbf{\Omega})$, where $N = \{s, 1, 2, \ldots, m, t\}$ is the set of nodes, $A = \{a_i \mid 1 \le i \le n\}$ is the set of arcs, and $\mathbf{\Omega} = \{c \mid c = (c_1, c_2, \ldots, c_n),\ c_i \in K_i \text{ for } 1 \le i \le n\}$ is the universal space of state vectors. The state (capacity) of arc a_i defined by c_i is a nonnegative integer random variable ranging from the smallest value 0 to the largest value u_i with a known probability distribution. A state vector $c = (c_1, c_2, \ldots, c_n)$ represents the current states of all arcs in the network. The state (capacity) of a network under state vector c is determined by the so-called structure function $M(c)$ denoting the maximum amount of flow that can be sent from the source node s to the sink node t when the sates of arcs are defined by c. For simplicity, the universal space $\mathbf{\Omega}$ can be denoted by its smallest state vector l and largest state vector u as $\mathbf{\Omega} = [l, u]$. Let C be a set of state vectors, then C is a subset of $\mathbf{\Omega}$.

For example, consider a multistate network $G(N, A, \mathbf{\Omega})$ shown in Figure 1 with $N = \{s, 1, 2, t\}$, $A = \{a_1, a_2, a_3, a_4, a_5, a_6\}$, and $\mathbf{\Omega} = [(0, 0, 0, 0, 0, 0), (3, 2, 1, 1, 1, 2)]$ according to Table 1. For a state vector $c = (2, 1, 1, 1, 1, 2)$, the state of the network under c is $M(c) = 3$, that is, the maximum amount of flow that can be sent from the source node s to the sink node t is 3 when $c_1 = 2$, $c_2 = 1$, $c_3 = 1$, $c_4 = 1$, $c_5 = 1$, $c_6 = 2$. Given a set of state vector $C = \{(c_1, c_2, c_3, c_4, c_5, c_6) \mid 1 \le c_1 \le 3,\ 1 \le c_2 \le 2,\ 0 \le c_3 \le 1,\ c_4 = 0,\ c_5 = 0,\ 0 \le c_6 \le 2\}$, its smallest state vector and largest state vector are $l^C = (1, 1, 0, 0, 0, 0)$ and $u^C = (3, 2, 1, 0, 0, 2)$, respectively.

Following the literature in [25, 26, 28–33], this paper considers four assumptions: (1) the state/capacity of each arc is a nonnegative integer random variable following a

Table 1: The states of arcs in Figure 1.

Arc	States				State probability distribution			
a_1	0	1	2	3	0.05	0.10	0.15	0.70
a_2	0	1	2	-	0.10	0.15	0.75	-
a_3	0	1	-	-	0.05	0.95	-	-
a_4	0	1	-	-	0.05	0.95	-	-
a_5	0	1	-	-	0.10	0.90	-	-
a_6	0	1	2	-	0.05	0.10	0.85	-

given probability distribution; (2) the states/capacities of different arcs are statistically independent; (3) each node is perfectly reliable; (4) all flows in the network obey the flow conservation law, i.e., total flows into and from a node (other than the source and sink nodes) are all equal.

2.2. Network Reliability. Reliability is a fundamental attribute for the safe operation of modern technological networks [34], and generally speaking, it represents the ability of a network to achieve its intended function within a given environment. For a multistate network, the reliability index R_d is defined as the probability that at least d units of flow can be sent from the source node to the sink node or the probability that the network capacity level is greater than or equal to a predetermined demand level d, that is $R_d = \Pr\{c \mid M(c) \geq d\}$. Evidently, to calculate R_d, one direct way is to enumerate every state vector c of $\mathbf{\Omega}$ by checking the condition $M(c) \geq d$, and then sum up their probabilities. And yet, it is extremely time-consuming due to the huge number of state vectors in $\mathbf{\Omega}$ for a multistate network. Instead, most of studies have focused on the d-minimal path method. A d-minimal path c is the minimal state vector satisfying $M(c) = d$. In other words, a state vector c is a d-minimal path if and only if $M(c) = d$, and $M(c - 0(a_i)) < d$ for each $c_i > 0$, where $0(a_i) = (0, \ldots, 0, 1, 0, \ldots, 0)$, i.e., the state of a_i is 1 and the states of other arcs are 0. Given that all d-minimal paths of a network are found, R_d can be calculated by the well-known Inclusion-Exclusion method or Sum of Disjoint Product method. Therefore, finding d-minimal paths is the key for computing R_d. In the following discussions, we focus on how to efficiently search for d-minimal paths in a multistate network.

3. Development of d-Minimal Path Algorithm

3.1. Determination of Search Space of d-Minimal Paths. Theoretically, the universal space $\mathbf{\Omega} = \{c \mid c = (c_1, c_2, \ldots, c_n),\ c_i \in K_i \text{ for } 1 \leq i \leq n\}$ is the whole search space of d-minimal paths, where $K_i = \{0, 1, \ldots, u_i\}$ is the set of states of a_i. It is a cumbersome task to search for d-minimal paths in $\mathbf{\Omega}$ due to the large number of state vectors. The definition of d-minimal path indicates that if a state vector c is a d-minimal path, two conditions must be met: $M(c) = d$, and $M(c - 0(a_i)) < d$ for each $c_i > 0$. As the source node s and the sink node t are two special nodes with the unique feature that s only has outgoing arcs and t only has incoming arcs, when c is a d-minimal path, the following conditions must be satisfied:

$$\sum_{a_i \in (s,\bullet)} c_i = d \quad \text{for } 0 \leq c_i \leq \min\{u_i, d\} \tag{1}$$

$$\sum_{a_j \in (\bullet,t)} c_j = d \quad \text{for } 0 \leq c_j \leq \min\{u_j, d\} \tag{2}$$

Now we define a new concept called qualified state vector to determine a smaller search space of d-minimal paths compared to $\mathbf{\Omega}$.

Definition 1. For a given demand level d, a state vector c is called a qualified state vector if c satisfies conditions (1) and (2).

The following theorem points out one basic property of qualified state vectors.

Theorem 2. *If c is a qualified state vector, then M(c) ≤ d.*

Proof. Note that $(s, \bullet)$ is a minimal cut of the network. Because $\sum_{a_i \in (s,\bullet)} c_i = d$, $M(c) \leq d$ follows from the max-flow min-cut theorem of network flows [35, 36].

Obviously, a d-minimal path is a qualified state vector, but a qualified state vector is not necessarily a d-minimal path. In conditions (1) and (2), suppose $(s,\bullet) = \{a_{s_1}, a_{s_2}, \ldots, a_{s_v}\}$ and $(\bullet, t) = \{a_{t_1}, a_{t_2}, \ldots, a_{t_w}\}$. Let $\delta^j = (c^j_{s_1}, c^j_{s_2}, \ldots, c^j_{s_v}, c^j_{t_1}, c^j_{t_2}, \ldots, c^j_{t_w})$ $(1 \leq j \leq H)$ represent a combination satisfying conditions (1) and (2), and $C^j = \{c \mid c_i = c^j_i \text{ for } a_i \in ((s,\bullet) \cup (\bullet, t)),\ 0 \leq c_i \leq \min\{u_i, d\} \text{ for } a_i \in A \setminus ((s,\bullet) \cup (\bullet, t))\}$ $(1 \leq j \leq H)$ is a set of state vectors corresponding to δ^j, then every state vector in C^j is a qualified state vector. Thus, $C^1 \cup C^2 \cup \ldots \cup C^H$ composes a new search space that is a subspace of the universal space $\mathbf{\Omega}$ and includes all qualified state vectors. Because a d-minimal path is a qualified state vector, the search space of d-minimal paths can be limited to $C^1 \cup C^2 \cup \ldots \cup C^H$ instead of $\mathbf{\Omega}$.

For illustration, we consider to find 3-minimal paths in Figure 1. Note that $(s, \bullet) = \{a_1, a_5\}$, and $(\bullet, t) = \{a_2, a_6\}$, then all possible combinations satisfying conditions (1) and (2) are $\delta^1 = (2, 1, 1, 2)$, $\delta^2 = (2, 1, 2, 1)$, $\delta^3 = (3, 0, 1, 2)$, and $\delta^4 = (3, 0, 2, 1)$. Thus, $C^1 = \{(c_1, c_2, c_3, c_4, c_5, c_6) \mid c_1 = 2,\ c_2 = 1,\ 0 \leq c_3 \leq 1,\ 0 \leq c_4 \leq 1,\ c_5 = 1,\ c_6 = 2\}$, $C^2 = \{(c_1, c_2, c_3, c_4, c_5, c_6) \mid c_1 = 2,\ c_2 = 2,\ 0 \leq c_3 \leq 1,\ 0 \leq c_4 \leq 1,\ c_5 = 1,\ c_6 = 1\}$,

$C^3 = \{(c_1, c_2, c_3, c_4, c_5, c_6) \mid c_1 = 3,\ c_2 = 1,\ 0 \le c_3 \le 1,\ 0 \le c_4 \le 1,\ c_5 = 0,\ c_6 = 2\}$, and $C^4 = \{(c_1, c_2, c_3, c_4, c_5, c_6) \mid c_1 = 3,\ c_2 = 2,\ 0 \le c_3 \le 1,\ 0 \le c_4 \le 1,\ c_5 = 0,\ c_6 = 1\}$. That is, all 3-minimal paths are included in search space $C^1 \cup C^2 \cup C^3 \cup C^4$. □

3.2. The Condition for Qualified State Vector to Be d-Minimal Path. When the amount of flow sent from s to t is d, a d-flow is a vector $(f_1^d, f_2^d, \ldots, f_n^d)$ consisting of flow f_i^d through each arc a_i $(1 \le i \le n)$ [31]. Thus, according to the theory of network flows [35, 36], a state vector c is a d-flow if and only if $M(c) = d$ and c satisfies the flow conservation law. The following theorem gives the condition for a qualified state vector to be a d-flow.

Theorem 3. *A qualified state vector c is a d-flow if and only if c satisfies the flow conservation law.*

Proof. If a qualified state vector c satisfies the flow conservation law, then c is a feasible flow; meanwhile, $\sum_{a_i \in (s,\bullet)} c_i = d$ and $\sum_{a_j \in (\bullet,t)} c_j = d$, which means $M(c) = d$. Hence, c is a d-flow.

A d-flow is not necessarily a d-minimal path, and their relationship is determined by Lemma 4 proposed by Xu and Niu [31]. □

Lemma 4. *A d-flow c is a d-minimal path if and only if there is no directed cycle in it.*

Thus, we can obtain the following conclusion.

Theorem 5. *A qualified state vector c is a d-minimal path if and only if c satisfies the flow conservation law and there is no directed cycle in it.*

Proof. Directly from Theorem 3 and Lemma 4.

Theorem 5 explicitly presents the sufficient and necessary condition for a qualified state vector to be d-minimal path. In the subsequent discussions, Theorem 5 will be utilized to search for d-minimal paths in set C^j $(1 \le j \le H)$. □

3.3. Searching for d-Minimal Paths in Set C^j $(1 \le j \le H)$. Recall that every state vector in C^j $(1 \le j \le H)$ is a qualified state vector, so the enumeration method, by Theorem 5, can be used to search for d-minimal paths in set C^j. However, it is still not practical to enumerate every state vector of C^j $(1 \le j \le H)$, and there is room for improvement. Next, we will detail how to integrate the max-flow algorithm with the enumeration method to search for d-minimal paths in set C^j $(1 \le j \le H)$.

For every set C^j $(1 \le j \le H)$, we consider its largest state vector u^{C^j}. If $M(u^{C^j}) < d$, then $M(c) \le M(u^{C^j}) < d$ for any state vector $c \in C$, which indicates there is no d-minimal path in C^j by definition, and thus C^j is discarded (i.e., there is no need to enumerate C^j). Consequently, only the case $M(u^{C^j}) = d$ needs to be discussed. The following theorem reveals that one d-flow can be derived from set C^j when $M(u^{C^j}) = d$.

Theorem 6. *For set C^j $(1 \le j \le H)$, if $M(u^{C^j}) = d$, then one d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ can be derived from C^j.*

Proof. Firstly, note that $u_i^{C^j} = l_i^{C^j}$ for $a_i \in ((s,\bullet) \cup (\bullet,t))$, and $l_i^{C^j} = 0$ for $a_i \in A \setminus ((s,\bullet) \cup (\bullet,t))$. Since $M(u^{C^j}) = d$, the maximum amount of flow that can be sent from s to t is d, and moreover the max-flow of the network under u^{C^j} is just a d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ according to the theory of network flows [35, 36]. As $f_i^d = u_i^{C^j} = l_i^{C^j}$ for $a_i \in ((s,\bullet) \cup (\bullet,t))$, $l_i^{C^j} = 0 \le f_i^d \le u_i^{C^j}$ for $a_i \in A \setminus ((s,\bullet) \cup (\bullet,t))$, then we have $l_i^{C^j} \le f_i^d \le u_i^{C^j}$ for $a_i \in A$.

In Theorem 6, the max-flow algorithm can be employed to compute $M(u^{C^j})$, and simultaneously determine a d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ when $M(u^{C^j}) = d$. By Lemma 4, if $(f_1^d, f_2^d, \ldots, f_n^d)$ contains no directed cycle, it is a d-minimal path. After finding a d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$, let $A^d(C^j) = \{a_i \mid a_i \in (A \setminus ((s,\bullet) \cup (\bullet,t))) \ \&\ f_i^d > l_i^{C^j}\}$. Note that $u_i^{C^j} = f_i^d = l_i^{C^j}$ for $a_i \in ((s,\bullet) \cup (\bullet,t))$, and $l_i^{C^j} \le f_i^d \le u_i^{C^j}$ for $a_i \in A \setminus ((s,\bullet) \cup (\bullet,t))$. If $A^d(C^j) = \Phi$, then $f_i^d = l_i^{C^j}$ for $a_i \in A \setminus ((s,\bullet) \cup (\bullet,t))$. As a result, we have $f_i^d = l_i^{C^j}$ for $a_i \in A$. That is, $l^{C^j} = (f_1^d, f_2^d, \ldots, f_n^d)$ when $A^d(C^j) = \Phi$. In such a case, as $M(l^{C^j}) = d$ and l^{C^j} is the smallest state vector in set C^j, it is impossible for any other state vector of C^j to be d-minimal path, and thus C^j is discarded.

If $A^d(C^j) \ne \Phi$, postulate that $A^d(C^j) = \{a_{x_1}, a_{x_2}, \ldots, a_{x_q}\}$, then we have $f_i^d > l_i^{C^j}$ for $a_i \in A^d(C^j)$, and $f_i^d = l_i^{C^j}$ for $a_i \in A \setminus A^d(C^j)$. Based on the obtained d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ and $A^d(C^j) = \{a_{x_1}, a_{x_2}, \ldots, a_{x_q}\}$, set C^j can be divided into disjoint subsets. The technique for dividing C^j is originally reported by Jane and Laih [23], and is adapted for use in this paper. Pivoting on arcs $a_{x_1}, a_{x_2}, \ldots, a_{x_q}$ one by one, C^j is divided into $C_1^j, C_2^j, \ldots, C_q^j$ and C_0^j as follows:

(1) Pivot on arc a_{x_1}:

$$C_1^j = \left\{(c_1, c_2, \ldots, c_n) \mid l_{x_1}^{C^j} \le c_{x_1} < f_{x_1}^d,\ c_i \in K_i^{C^j} \text{ for } a_i \in A \setminus \left\{a_{x_1}\right\}\right\}; \tag{3}$$

(2) Pivot on arc a_{x_2}:

$$C_2^j = \left\{(c_1, c_2, \ldots, c_n) \mid f_{x_1}^d \le c_{x_1} \le u_{x_1}^{C^j},\ l_{x_2}^{C^j} \le c_{x_2} < f_{x_2}^d,\ c_i \in K_i^{C^j} \text{ for } a_i \in A \setminus \left\{a_{x_1}, a_{x_2}\right\}\right\}; \tag{4}$$

...

(q) Pivot on arc a_{x_q}:

$$C_q^j = \left\{(c_1, c_2, \ldots, c_n) \mid f_i^d \le c_i \le u_i^{C^j} \text{ for } a_i \in \left\{a_{x_1}, a_{x_2}, \ldots, a_{x_{q-1}}\right\},\ l_{x_q}^{C^j} \le c_{x_q} < f_{x_q}^d,\ c_i \in K_i^{C^j} \text{ for } a_i \in A \setminus A^d\left(C^j\right)\right\}; \quad (5)$$

(q+1)

$$C_0^j = \left\{(c_1, c_2, \ldots, c_n) \mid f_i^d \le c_i \le u_i^{C^j} \text{ for } a_i \in A^d\left(C^j\right),\ c_i \in K_i^{C^j} \text{ for } a_i \in A \setminus A^d\left(C^j\right)\right\}. \quad (6)$$

□

According to Jane and Laih [23], subsets C_0^j, C_1^j, C_2^j,..., and C_q^j satisfy two properties: (1) $C^j = C_0^j \cup C_1^j \cup C_2^j \cup \ldots \cup C_q^j$; (2) $C_0^j, C_1^j, C_2^j, \cdots$, and C_q^j are disjoint. Meanwhile, it is obvious that all of the state vectors in C_k^j $(0 \le k \le q)$ are still qualified state vectors.

The remaining work is to search for d-minimal paths in these subsets. First, we consider C_0^j, and note that the d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ derived from C^j is the smallest state vector in C_0^j, then all of the other state vectors in C_0^j are no d-minimal paths by definition. In such a case, there is no need to enumerate the state vectors in C_0^j, i.e., C_0^j is discarded.

Now we consider C_k^j for $2 \le k \le q$. If $M(u^{C_k^j}) < d$, then there exists no d-minimal path in C_k^j, and C_k^j is discarded either. But, if $M(u^{C_k^j}) = d$, the existence of d-minimal paths in C_k^j is uncertain. In this case, because all of the state vectors in C_k^j $(0 \le k \le q)$ are still qualified state vectors, by Theorem 5, the enumeration method can be used to search for d-minimal paths in C_k^j.

Finally, we consider C_1^j. If $M(u^{C_1^j}) < d$, then there exists no d-minimal path in C_1^j, and C_1^j is discarded. Now we discuss the case $M(u^{C_1^j}) = d$. Observe that $C_1^j = \{(c_1, c_2, \ldots, c_n) \mid l_{x_1}^{C^j} \le c_{x_1} < f_{x_1}^d,\ c_i \in K_i^{C^j} \text{ for } a_i \in A \setminus \{a_{x_1}\}\}$ which means $l^{C_1^j} = l^{C^j}$, then C_1^j can be coped with in the same way as C^j. That is, a d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ can be obtained from C_1^j when $M(C_1^j) = d$ by Theorem 6, and then C_1^j is similarly decomposed into disjoint subsets that are discussed in the same way as the subsets derived from C^j. This decomposition process will stop when there exists no subset $C*$ meeting $l^{C*} = l^{C^j}$ and $M(u^{C*}) = d$. At this moment, all of the d-minimal paths in set C^j have been found.

3.4. The Developed Algorithm. According to the above discussions, we now present an algorithm for finding d-minimal paths in a multistate network $G(N, A, \mathbf{\Omega})$. To make the proposed algorithm more understandable, we use two parts, Algorithm A and Algorithm B, to describe it hereunder.

Algorithm A. Finding d-minimal paths in a multistate network $G(N, A, \mathbf{\Omega})$.

Input. A multistate network $G(N, A, \mathbf{\Omega})$ with demand level d.

Output. All d-minimal paths.

A. Step 1. Use the enumeration method to solve all of the combinations satisfying conditions (1) and (2), and then determine the search space $C^1 \cup C^2 \cup \ldots \cup C^H$.

A. Step 2. For each $C^j (1 \le j \le H)$, let $C = C^i$ and call *Algorithm B* to search for d-minimal paths in C.

Algorithm B. Finding d-minimal paths in set C.

B. Step 1. Compute $M(u^C)$. If $M(u^C) < d$, there is no d-minimal path in C, and stop.

B. Step 2. Find a d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ in C, if it contains no directed cycle, it is a d-minimal path.

B. Step 3. Compute $A^d(C) = \{a_i \mid a_i \in A \setminus ((s,\bullet) \cup (\bullet, t))\ \&\ f_i^d > l_i^{C^j}\}$. If $A^d(C) = \Phi$, then stop; otherwise, suppose $A^d(C) = \{a_{x_1}, a_{x_2}, \ldots, a_{x_q}\}$.

B. Step 4. Divide C into q+1 nonempty disjoint subsets $C_0, C_1, C_2, \ldots, C_q$.

B. Step 5. If $q > 1$, compute $M(u^{C_k})$ $(2 \le k \le q)$. If $M(u^{C_k}) < d$, there is no d-minimal path in C_k; otherwise, for each $c \in C_k$ $(2 \le k \le q)$, if c satisfies the flow conservation law and there is no directed cycle in it, it is a d-minimal path.

B. Step 6. Let $C = C_1$, and go to *B. Step 1.*

To help follow the procedure of the proposed algorithm, we briefly explain the role of every step. In Algorithm A, A. Sep 1 is a proprocessing step for determining a smaller search space of d-minimal paths, which includes H sets, and A. Step 2 is to search for d-minimal paths of each set C^j by calling Algorithm B. Note that Algorithm B is a recursive algorithm for solving d-minimal paths. B. Step 1 checks whether there may exist d-minimal paths in C by computing the maximum flow under the largest state vector. B. Step 2 is to find a d-flow and to check whether it is a d-minimal path. Once a d-flow is obtained, B. Step 3 and B. Step 4 are to divide set C into q+1 disjoint subsets. B. Step 5 searches for d-minimal paths in subsets $C_2, \ldots, C_q$. Subset C_1 is regarded as a new set to be solved from the beginning step, i.e., B. Step 1 (a new iteration), and thus B. Step 6 is to put subset C_1 into the next iteration.

It is noteworthy that the major characteristics of the proposed algorithm are twofold. First, a smaller search space of d-minimal paths is determined in A. Step 1. Second, the search space is recursively divided into subspaces such that the traditional max-flow algorithm (find a d-flow in B. Step 2) and the enumeration method (check whether c satisfies the flow conservation law in B. Step 5) are integrated to search

for d-minimal paths in subsets. Notably, there is no need to enumerate subset C_0, and when $M(u^{C_k}) < d$ $(2 \le k \le q)$ holds in B. Step 5, there is also no need to enumerate subset C_k (no d-minimal paths exist in C_k either), which no doubt contributes to the efficiency improvement in searching for d-minimal paths.

Now, we discuss the time complexity of the proposed algorithm. Complexity analysis of Algorithm A: it takes $O(n(\prod_{a_i \in (s,\bullet)}(1+\min\{u_i,d\}) + \prod_{a_j \in (\bullet,t)}(1+\min\{u_j,d\})))$ time to solve all of the combinations satisfying conditions (1) and (2). After obtaining the combinations, it then needs $O(Hn)$ time to determine the search space. As a result, A. Step 1 totally requires $O(n(\prod_{a_i \in (s,\bullet)}(1+\min\{u_i,d\}) + \prod_{a_j \in (\bullet,t)}(1+\min\{u_j,d\})))$ time. The time complexity of A. Step 2 is obviously determined by the time complexity of Algorithm B. Complexity analysis of Algorithm B: B. Step 1 takes $O(mn^2)$ time to compute $M(u^C)$ [37]. Once $M(u^C) = d$ is determined, a d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ can be obtained in $O(n)$ time, and checking a directed cycle needs $O(m)$ time. Thus, B. Step 2 requires $O(m+n)$ time to check whether d-flow $(f_1^d, f_2^d, \ldots, f_n^d)$ is a d-minimal path or not. B. Step 3 requires $O(n)$ time to compute $A^d(C)$. B. Step 4 takes $O(n)$ time to divide set C. B. Step 6 takes $O(n)$ time to implement assignment operation. Therefore, the algorithm totally takes $O(mn^2)$ time without consideration of the time complexity of B. Step 5. Since solving all d-minimal paths is an NP-hard problem, the time complexity of the proposed algorithm mainly depends on B. Step 5 that employs the enumeration method to search for d-minimal paths. The number of enumerated state vectors in B. Step 5 is upper bounded by $|C^1 \cup C^2 \cup \ldots \cup C^H|$, i.e., $H\prod_{a_i \in (A\setminus((s,\bullet)\cup(\bullet,t)))}(1+\min\{u_i,d\})$, and it takes $O(m)$ time to check whether a qualified state vector satisfies the flow conservation law, and $O(m)$ time to check a directed cycle. Thus, B. Step 5 takes $O(mH\prod_{a_i \in (A\setminus((s,\bullet)\cup(\bullet,t)))}(1+\min\{u_i,d\}))$ time in the worst case. Thus, the time complexity of Algorithm B is $O(mH\prod_{a_i \in (A\setminus((s,\bullet)\cup(\bullet,t)))}(1+\min\{u_i,d\}))$ in the worst case. As a result, the time complexity of the proposed algorithm is $O(mH\prod_{a_i \in (A\setminus((s,\bullet)\cup(\bullet,t)))}(1+\min\{u_i,d\}))$.

We compare the proposed algorithm with the existing algorithms theoretically in terms of time complexity. Note that the algorithms in [13, 27, 29] are all based on the model proposed by Lin et al. [25], so they have the same time complexity as the algorithm by Lin et al. [25]. The time complexity of the algorithm by Lin et al. [25] is $O\left(mp\min\left\{\binom{p+d-1}{d}, \prod_{i=1}^{p}(1+\min\{CP_i,d\})\right\}\right)$ where p is the number of minimal paths and $O(p) = O(2^m)$, and the time complexity of both the algorithm by Yeh [26] and the modified algorithm by Xu and Niu [31] is $O((m+n)\prod_{a_i \in A}(1+\min\{u_i,d\}))$. As H is the total number of possible combinations satisfying conditions (1) and (2), it is trivial to have $H < \prod_{a_i \in (s,\bullet)\cup(\bullet,t)}(1+\min\{u_i,d\})$, and thus $O(mH\prod_{a_i \in (A\setminus((s,\bullet)\cup(\bullet,t)))}(1+\min\{u_i,d\})) \le O(m\prod_{a_i \in A}(1+\min\{u_i,d\})) \le O((m+n)\prod_{a_i \in A}(1+\min\{u_i,d\}))$. Furthermore, notice that $O((m+n)\prod_{a_i \in A}(1+\min\{u_i,d\})) \ll O\left(\left(mp\min\left\{\binom{p+d-1}{d}, \prod_{i=1}^{p}(1+\min\{CP_i,d\})\right\}\right)\right)$ [26]. Therefore, we can obtain $O(mH\prod_{a_i \in (A\setminus((s,\bullet)\cup(\bullet,t)))}(1+\min\{u_i,d\})) \le O((m+n)\prod_{a_i \in A}(1+\min\{u_i,d\})) \ll O\left(mp\min\left\{\binom{p+d-1}{d}, \prod_{i=1}^{p}(1+\min\{CP_i,d\})\right\}\right)$. That is, the proposed algorithm compares favorably with the algorithms of Lin et al. [25], Yeh [26], and Xu and Niu [31] in terms of time complexity.

4. Computational Examples

4.1. Illustrative Example. The multistate network in Figure 1 is adopted to elucidate the proposed algorithm. Given the demand level $d = 3$, the following steps illustrate how to search for all of 3-minimal paths using the proposed algorithm (For simplicity, a set of state vectors C is denoted by its smallest state vector l^C and largest state vector u^C as $[l^C, u^C]$).

Algorithm A.

A. Step 1. Use the enumeration method to solve all of the combinations satisfying conditions (1) and (2) as follows:

$$c_1 + c_5 = 3, \text{ where } 0 \le c_1 \le 3 \text{ and } 0 \le c_5 \le 1$$
$$c_2 + c_6 = 3, \text{ where } 0 \le c_2 \le 2 \text{ and } 0 \le c_6 \le 2$$

And the obtained combinations are (2, 1, 1, 2), (2, 1, 2, 1), (3, 0, 1, 2), and (3, 0, 2, 1), Thus, the search spaces are $C^1 = [(2, 1, 0, 0, 1, 2), (2, 1, 1, 1, 1, 2)]$, $C^2 = [(2, 2, 0, 0, 1, 1), (2, 2, 1, 1, 1, 1)]$, $C^3 = [(3, 1, 0, 0, 0, 2), (3, 1, 1, 1, 0, 2)]$ and $C^4 = [(3, 2, 0, 0, 0, 1), (3, 2, 1, 1, 0, 1)]$.

A. Step 2. Let $C = C^1 = [(2, 1, 0, 0, 1, 2), (2, 1, 1, 1, 1, 2)]$ and search for 3-minimal paths by calling Algorithm B as follows:

B. Step 1. $M(u^C) = 3$.

B. Step 2. (2, 1, 1, 0, 1, 2) is a 3-flow in C, and it contains no directed cycle, then (2, 1, 1, 0, 1, 2) is a 3-minimal path.

B. Step 3. $A^3(C) = \{a_3\} \ne \Phi$, $q = 1$.

B. Step 4. C is divided into two subsets $C_0 = [(2, 1, 1, 0, 1, 2), (2, 1, 1, 1, 1, 2)]$ and $C_1 = [(2, 1, 0, 0, 1, 2), (2, 1, 0, 1, 1, 2)]$.

B. Step 5. $q = 1$.

B. Step 6. Let $C = C_1 = [(2, 1, 0, 0, 1, 2), (2, 1, 0, 1, 1, 2)]$, and go to B. Step 1. // next iteration //

B. Step 1. $M(u^C) = 2 < 3$, then there is no 3-minimal path in C, and stop.

A. Step 2. Let $C = C^2 = [(2, 2, 0, 0, 1, 1), (2, 2, 1, 1, 1, 1)]$ and search for 3-minimal paths by calling Algorithm B as follows:

B. Step 1. $M(u^C) = 3$.

$\cdots$

Finally, there are totally obtained three 3-minimal paths, as shown in Table 2. Let $V_1 = \{c \mid c \ge (2, 1, 1, 0, 1, 2)\}$, $V_2 = \{c \mid c \ge (2, 2, 0, 0, 1, 1)\}$, $V_3 = \{c \mid c \ge (3, 2, 1, 0, 0, 1)\}$, then

Table 2: The obtained results using the proposed algorithm.

Search space	3-minimal path
$C^1 = [(2,1,0,0,1,2)],(2,1,1,1,1,2)]$	(2, 1, 1, 0, 1, 2)
$C^2 = [(2,2,0,0,1,1)],(2,2,1,1,1,1)]$	(2, 2, 0, 0, 1, 1)
$C^3 = [(3,1,0,0,0,2)],(3,1,1,1,0,2)]$	-
$C^4 = [(3,2,0,0,0,1)],(3,2,1,1,0,1)]$	(3, 2, 1, 0, 0, 1)

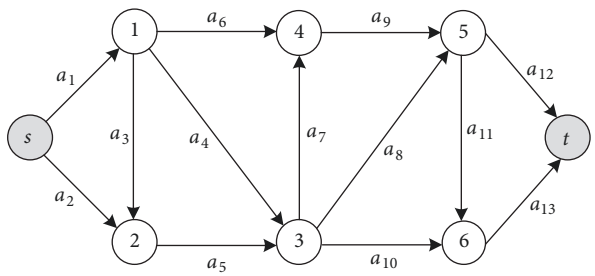

Figure 2: A benchmark network for numerical experiments.

R_3 can be evaluated via the Inclusion-Exclusion method as follows: $R_3 = \Pr\{c \mid M(c) \geq d\} = \Pr(V_1 \cup V_2 \cup \ldots \cup V_3) = \Pr\{V_1\} + \Pr\{V_2\} + \Pr\{V_3\} - \Pr\{V_1 \cap V_2\} - \Pr\{V_1 \cap V_3\} - \Pr\{V_2 \cap V_3\} + \Pr\{V_1 \cap V_2 \cap V_3\} = \Pr\{x \mid x \geq (2,1,1,0,1,2)\} + \Pr\{x \mid x \geq (2,2,0,0,1,1)\} + \Pr\{x \mid x \geq (3,2,1,0,0,1)\} - \Pr\{x \mid x \geq (2,2,1,0,1,2)\} - \Pr\{x \mid x \geq (3,2,1,0,1,1)\} - \Pr\{x \mid x \geq (3,2,1,0,1,2)\} + \Pr\{x \mid x \geq (3,2,1,0,1,2)\} = 0.685104$.

4.2. Numerical Example for Efficiency Investigation. In this section, we focus on the computational efficiency of the proposed algorithm in finding all d-minimal paths. As mentioned before, the method in [31] has been proved to be more efficient than the algorithms of Lin et al. [25], and Yeh [26] in terms of finding d-minimal paths, so we compare the proposed algorithm with it through numerical computations. Both algorithms are coded in a MATLAB program, and run on the same personal computer with Intel(R) Core (TM) i5-3210M 2.50 GHz CPU.

A medium-sized network shown in Figure 2 is used as benchmark network to perform numerical experiments. For comparison, we consider two scenarios. Scenario 1: the largest states of all arcs are equal to 4, i.e., $u_i = 4$ $(1 \leq i \leq 13)$, and Scenario 2: the largest states of all arcs are equal to 5, i.e., $u_i = 5$ $(1 \leq i \leq 13)$. For each scenario, the demand level d ranges from 1 to the largest value $M(u)$, i.e., all of the d-minimal paths corresponding to multiple demand levels are solved. We focus on the computational time required to search for all d-minimal paths of each demand level. Meanwhile, to make intuitive comparisons, we define ratio λ as the computational time of the method in [31] divided by the computational time of the proposed algorithm. Then, the ratio λ represents the relative efficiency of two algorithms.

From both Figures 3 and 4, it can be seen that the ratio λ is always above one for every demand level d under both scenarios, implying the proposed algorithm is more efficient than the method in [31]. Meanwhile, note that as the demand level d increases, the ratio λ increases, indicating

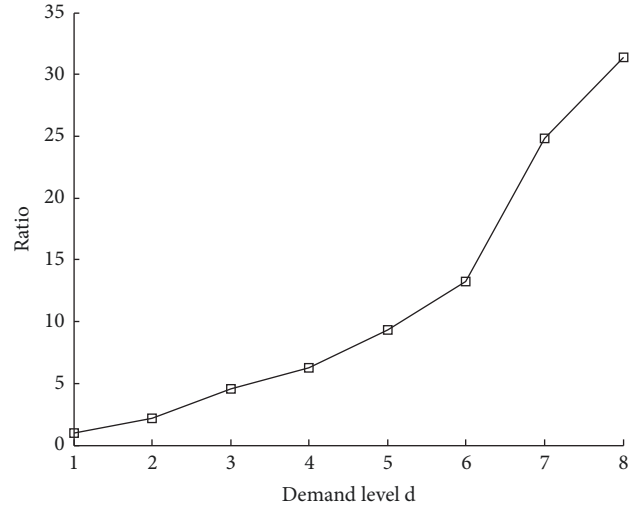

Figure 3: Ratio λ when $u_i = 4$.

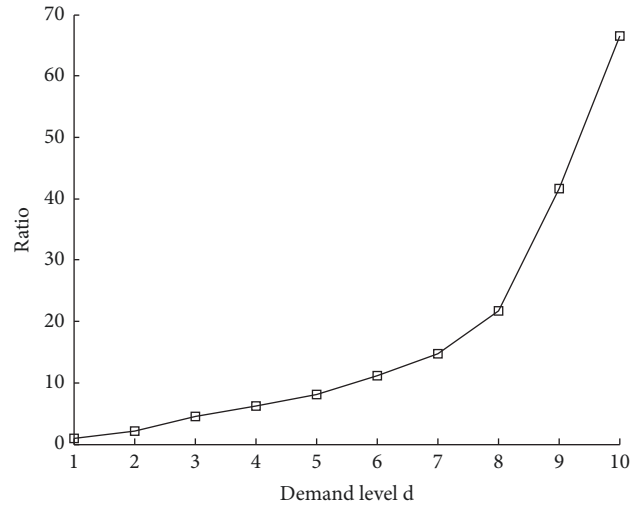

Figure 4: Ratio λ when $u_i = 5$.

the efficiency advantage of the proposed algorithms becomes much stronger. For example, the proposed algorithm is more than 10 times faster than the algorithm in [31] when solving d-minimal paths with demand level d above 5 under both Scenario 1 and Scenario 2, and the proposed algorithm is more than 20 times faster than the algorithm in [31] when solving d-minimal paths with demand level d above 6 under Scenario 1 and with demand level d above 7 under Scenario 2, respectively.

5. Case Study of Power Transmission Network

In modern society, electric power is the necessity to our daily life. Thus, a reliable power transmission network is critical to the stable operation of the whole society. As exemplified in Section 1, power transmission networks can be regarded as typical multistate networks. Therefore, we take a power transmission network as case study to demonstrate the utility

Table 3: Data of each arc (transmission line) of the power transmission network.

Arc	Capacity (MW)					
	0	1	2	3	4	5
	Capacity probability distribution					
a_1	0.005	0.008	0.012	0.016	0.022	0.937
a_2	0.006	0.023	0.971	-	-	-
a_3	0.004	0.018	0.026	0.952	-	-
a_4	0.005	0.011	0.023	0.039	0.922	-
a_5	0.006	0.029	0.965	-	-	-
a_6	0.004	0.018	0.025	0.953	-	-
a_7	0.005	0.016	0.026	0.953	-	-
a_8	0.007	0.019	0.974	-	-	-
a_9	0.006	0.015	0.979	-	-	-
a_{10}	0.004	0.013	0.983	-	-	-
a_{11}	0.008	0.013	0.026	0.953	-	-
a_{12}	0.007	0.026	0.967	-	-	-
a_{13}	0.006	0.019	0.034	0.941	-	-
a_{14}	0.007	0.012	0.029	0.952	-	-
a_{15}	0.009	0.016	0.029	0.035	0.911	-
a_{16}	0.003	0.008	0.012	0.016	0.022	0.939

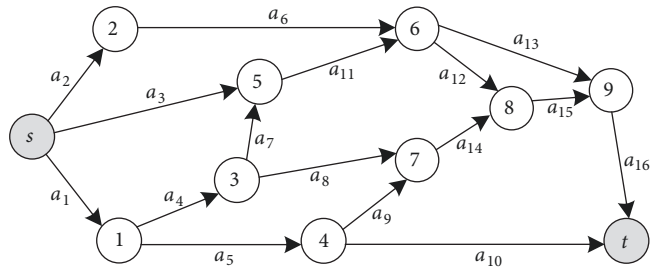

Figure 5: A power transmission network.

of the proposed algorithm and the related implications of network reliability.

A power transmission network with sixteen transmission lines is shown in Figure 5. The capacity and capacity distribution of each transmission line of the power transmission network are presented in Table 3. Suppose that the supervisor would like to assess the capability of the power transmission network to successfully transmit 6 units (MW) of electric power from the power plant (node s) to the targeted high voltage substation (node t). Then, the network reliability R_6 can be computed in terms of 6-minimal paths. Using the proposed algorithm, a total of 143 6-minimal paths are obtained. As a result, it can be calculated that R_6 = 0.928546 by the Inclusion-Exclusion method. That is, the probability of the power transmission network being able to successfully transmit 6 units (MW) of electric power from the power plant to the targeted high voltage substation is 0.928546. This value 0.928546 provides the supervisor with intuitive information regarding the capability of the network to ensure the transmission of 6 units (MW) of electric power from the power plant to the targeted high voltage substation. If 0.928546 is above the supervisor's expectation, it means the operational state of the power transmission network is desirable. However, if 0.928546 is below the supervisor's expectation, it means the operational state of the power transmission network is undesirable, and then some improvement measures are needed to enhance the power transmission network.

6. Conclusion

In this paper, an efficient algorithm is developed to search for d-minimal paths in reliability evaluation of multistate networks. Unlike the existing algorithms that enumerate d-minimal paths in a larger search space, the proposed algorithm first determines a relatively smaller search space of d-minimal paths by employing a concept of qualified state vector. A sufficient and necessary condition for a qualified state vector to be d-minimal path is derived. And, the search space is recursively divided into subspaces, so that the max-flow algorithm and the enumeration algorithm are integrated to search for d-minimal paths with the purpose of increasing the efficiency of enumeration. Furthermore, it is theoretically proven that the proposed algorithm compares favorably

with the existing algorithms in terms of time complexity. Finally, numerical examples have shown the effectiveness and efficiency of the proposed algorithm.

Notations

$G(N, A, \Omega)$: A multistate network with a set of nodes $N = \{s, 1, 2, \ldots, m, t\}$, a set of arcs $A = \{a_i \mid 1 \le i \le n\}$, and the universal space of state vectors $\Omega = \{c \mid c = (c_1, c_2, \ldots, c_n),\ c_i \in K_i \text{ for } 1 \le i \le n\}$
s/t: The source node/the sink node
m/n: The number of nodes except s and t/the number of arcs
a_i: ith arc in A
c_i: A state of arc a_i
c: A state vector
C: A set of state vector
K_i: The set of states of arc a_i, i.e., $K_i = \{0, 1, \ldots, u_i\}$
l_i/u_i: The smallest/largest state of arc a_i
l/u: The smallest/largest state vector
l^C/u^C: The smallest/largest state vector with respect to set C
l_i^C/u_i^C: The smallest/largest state of arc a_i with respect to set C
K_i^C: The set of states of arc a_i with respect to C
$M(c)$: Maximum amount of flow that can be sent from s to t when the network is under state vector c
d: Demand level
R_d: Network reliability at demand level d
$0(a_i)$: A special state vector with the state of a_i being 1 and the states of other arcs being 0, i.e., $0(a_i) = (0, \ldots, 0, 1, 0, \ldots, 0)$
$(s, \bullet)$: The set of outgoing arcs of s
$(\bullet, t)$: The set of incoming arcs of t
δ^j: jth combination satisfying conditions (1) and (2)
C^j: jth set of state vectors corresponding to δ^j
C_i^j: ith subset derived from the decomposition of set C^j
H: The total number of combination satisfying conditions (1) and (2)
p: The number of minimal paths
CP_i: The capacity of ith minimal path
f_i^d: The flow through arc a_i when d unit of flow is sent from s to t
$|\bullet|$: The number of elements in $\bullet$.

Data Availability

All data are provided in full in the results section of this paper.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is mainly supported by the National Natural Science Foundation of China (Projects nos. 71601072 and 61872126), the Research Project from the Science and Technology Department of Henan Province (Project no. 172102310677), and the Fundamental Research Funds for the Universities of Henan Province (Project no. NSFRF170914).

References

[1] Y.-K. Lin and P.-C. Chang, "Estimated and exact system reliabilities of a maintainable computer network," *Journal of Systems Science and Systems Engineering*, vol. 20, no. 2, pp. 229–248, 2011.

[2] W.-C. Yeh, "Evaluation of the one-to-all-target-subsets reliability of a novel deterioration-effect acyclic multi-state information network," *Reliability Engineering & System Safety*, vol. 166, pp. 132–137, 2017.

[3] Y.-K. Lin and C.-T. Yeh, "Maximal network reliability for a stochastic power transmission network," *Reliability Engineering & System Safety*, vol. 96, no. 10, pp. 1332–1339, 2011.

[4] M. López-Campos, F. Kristjanpoller, P. Viveros, and R. Pascual, "Reliability assessment methodology for massive manufacturing using multi-function equipment," *Complexity*, vol. 2018, Article ID 4084917, 8 pages, 2018.

[5] S. Zhang, X. Li, and C. Zhang, "A fuzzy control model for restraint of bullwhip effect in uncertain closed-loop supply chain with hybrid recycling channels," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 2, pp. 475–482, 2017.

[6] S. Zhang, Y. Hou, S. Zhang, and M. Zhang, "Fuzzy control model and simulation for nonlinear supply chain system with lead times," *Complexity*, vol. 2017, Article ID 2017634, 11 pages, 2017.

[7] Y.-K. Lin, C.-T. Yeh, and C.-F. Huang, "Reliability assessment of a multistate freight network for perishable merchandise with multiple suppliers and buyers," *International Journal of Systems Science*, vol. 48, no. 1, pp. 74–83, 2017.

[8] C.-T. Yeh, Y.-K. Lin, and C.-F. Huang, "A reliability indicator to measure a stochastic supply chain network with transportation damage and limited production capacity," *IIE Transactions*, vol. 46, no. 10, pp. 1066–1078, 2014.

[9] C.-C. Jane, "Performance evaluation of logistics systems under cost and reliability considerations," *Transportation Research Part E: Logistics and Transportation Review*, vol. 47, no. 2, pp. 130–137, 2011.

[10] C.-C. Jane and Y.-W. Laih, "Evaluating cost and reliability integrated performance of stochastic logistics systems," *Naval Research Logistics (NRL)*, vol. 59, no. 7, pp. 577–586, 2012.

[11] Y.-F. Niu, Z.-Y. Gao, and W. H. K. Lam, "Evaluating the reliability of a stochastic distribution network in terms of minimal cuts," *Transportation Research Part E: Logistics and Transportation Review*, vol. 100, pp. 75–97, 2017.

[12] Y.-F. Niu, W. H. K. Lam, and Z. Gao, "An efficient algorithm for evaluating logistics network reliability subject to distribution cost," *Transportation Research Part E: Logistics and Transportation Review*, vol. 67, pp. 175–189, 2014.

[13] X. Xu, Y. Niu, and Q. Li, "Performance assessment of a freight network with stochastic capacities," *Complexity*, vol. 2018, Article ID 9142542, 9 pages, 2018.

[14] Y. Zhang, K. Lv, S. Wang, J. Su, and D. Meng, "Modeling gene networks in saccharomyces cerevisiae based on gene expression profiles," *Computational and Mathematical Methods in Medicine*, vol. 2015, Article ID 621264, 10 pages, 2015.

[15] Y. Zhang, M. Zhao, J. Su, X. Lu, and K. Lv, "Novel model for cascading failure based on degree strength and its application in directed gene logic networks," *Computational and Mathematical Methods in Medicine*, vol. 2018, Article ID 8950794, 9 pages, 2018.

[16] S. Wang, Y. Chen, Q. Wang, E. Li, Y. Su, and D. Meng, "Analysis for gene networks based on logic relationships," *Journal of Systems Science & Complexity*, vol. 23, no. 5, pp. 999–1011, 2010.

[17] Y.-F. Niu and F.-M. Shao, "A practical bounding algorithm for computing two-terminal reliability based on decomposition technique," *Computers & Mathematics with Applications*, vol. 61, no. 8, pp. 2241–2246, 2011.

[18] R. Mishra, M. A. Saifi, and S. K. Chaturvedi, "Enumeration of minimal cutsets for directed networks with comparative reliability study for paths or cuts," *Quality and Reliability Engineering International*, vol. 32, no. 2, pp. 555–565, 2016.

[19] P. Caşcaval, "Approximate method to evaluate reliability of complex networks," *Complexity*, vol. 2018, Article ID 5967604, 11 pages, 2018.

[20] S. Zhu, J. Lou, Y. Liu, Y. Li, and Z. Wang, "Event-triggered control for the stabilization of probabilistic boolean control networks," *Complexity*, vol. 2018, Article ID 9259348, 7 pages, 2018.

[21] M. P. Xing, H. Shen, and Z. Wang, "H_∞ synchronization of semi-Markovian jump neural networks with randomly occurring time-varying delays," *Complexity*, vol. 2018, Article ID 8094292, 16 pages, 2018.

[22] C. Alexopoulos, "A note on state-space decomposition methods for analyzing stochastic flow networks," *IEEE Transactions on Reliability*, vol. 44, no. 2, pp. 354–357, 1995.

[23] C.-C. Jane and Y.-W. Laih, "A practical algorithm for computing multi-state two-terminal reliability," *IEEE Transactions on Reliability*, vol. 57, no. 2, pp. 295–302, 2008.

[24] C.-C. Jane and Y.-W. Laih, "Computing multi-state two-terminal reliability through critical arc states that interrupt demand," *IEEE Transactions on Reliability*, vol. 59, no. 2, pp. 338–345, 2010.

[25] J. Lin, C. Jane, and J. Yuan, "On reliability evaluation of a capacitated-flow network in terms of minimal pathsets," *Networks*, vol. 25, no. 3, pp. 131–138, 1995.

[26] W.-C. Yeh, "A novel method for the network reliability in terms of capacitated-minimum-paths without knowing minimum-paths in advance," *Journal of the Operational Research Society*, vol. 56, no. 10, pp. 1235–1240, 2005.

[27] Y.-K. Lin, "A simple algorithm for reliability evaluation of a stochastic-flow network with node failure," *Computers & Operations Research*, vol. 28, no. 13, pp. 1277–1285, 2001.

[28] W.-C. Yeh, "A simple method to verify all d-minimal path candidates of a limited-flow network and its reliability," *The International Journal of Advanced Manufacturing Technology*, vol. 20, no. 1, pp. 77–81, 2002.

[29] S.-G. Chen and Y.-K. Lin, "Searching for d-MPs with fast enumeration," *Journal of Computational Science*, vol. 17, pp. 139–147, 2016.

[30] Y. Niu, Z. Gao, and H. Sun, "An improved algorithm for solving all d-MPs in multi-state networks," *Journal of Systems Science and Systems Engineering*, vol. 26, no. 6, pp. 711–731, 2017.

[31] X. Z. Xu and Y. F. Niu, "A hybrid algorithm for reliability evaluation of a multi-state system," *Journal of The Chinese Institute of Engineers*, vol. 36, no. 2, pp. 173–179, 2013.

[32] G. Bai, M. J. Zuo, and Z. Tian, "Search for all d-MPs for all d levels in multistate two-terminal networks," *Reliability Engineering & System Safety*, vol. 142, pp. 300–309, 2015.

[33] W.-C. Yeh, "Fast algorithm for searching d-MPs for all possible d," *IEEE Transactions on Reliability*, vol. 67, no. 1, pp. 308–315, 2018.

[34] E. Zio, "Reliability engineering: old problems and new challenges," *Reliability Engineering & System Safety*, vol. 94, no. 2, pp. 125–141, 2009.

[35] J. Ford and D. R. Fulkerson, *Flows in Networks*, Princeton University Pres, Princeton, NJ, USA, 1962.

[36] R. K. Ahuja, T. L. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1993.

[37] R. K. Ahuja, M. Kodialam, A. K. Mishra, and J. B. Orlin, "Computational investigations of maximum flow algorithms," *European Journal of Operational Research*, vol. 97, no. 3, pp. 509–542, 1997.