

# A generalized multistate-based path vector approach to multistate two-terminal reliability

JOSE E. RAMIREZ-MARQUEZ<sup>1,\*</sup>, DAVID W. COIT<sup>2</sup> and MICHAEL TORTORELLA<sup>2</sup>

<sup>1</sup>*Department of Systems Engineering and Engineering Management, Stevens Institute of Technology, Hoboken, NJ 07030, USA*  
E-mail: jmarquez@stevens.edu

<sup>2</sup>*Department of Industrial & Systems Engineering, Rutgers University, Piscataway, NJ 08854, USA*

Received December 2004 and accepted August 2005

---

The two-terminal reliability problem assumes that a network and its elements are either in a working or a failed state. However, many practical networks are built of elements that may operate in more than two states i.e., elements may be degraded but still functional. Multistate two-terminal reliability at demand level  $d$  (M2TR <sub>$d$</sub> ) can be defined as the probability that the system capacity generated by multistate components is greater than or equal to a demand of  $d$  units. This paper presents a fully multistate-based algorithm that obtains the multistate equivalent of binary path sets, namely, Multistate Minimal Path Vectors (MMPVs), for the M2TR <sub>$d$</sub>  problem. The algorithm mimics natural organisms in the sense that a select number of arcs inherit information from other specific arcs contained in a special set called the “primary set.” The algorithm is tested and compared with published results in the literature. Two features of the algorithm make it relevant: (i) unlike other approaches, it does not depend on an *a priori* knowledge of the binary path sets to obtain the MMPVs; and (ii) the use of an information sharing approach and network reduction technique significantly reduce the number of vector analyses needed to obtain all the component levels that guarantee system success. Additionally, the complexities associated with the computation of reliability are discussed. A Monte Carlo simulation approach is used to obtain an accurate estimate of actual M2TR values based on MMPVs. Examples are used to validate the algorithm and the simulation procedure.

## 1. Introduction

System-reliability evaluation can be understood as the computation of the probability that a system is functioning for a given amount of time under specified conditions (Elsayed, 1996). In the mathematical theory of reliability, the evaluation of this probability can be broadly defined as a two-phase problem. The first phase consists in analyzing how a system’s operation is affected by the behavior of its components. This is a crucial phase because it requires an understanding of the interactions between the components and the system behavior. The second phase is computational. Based on the interactions between the components and the system and the quantitative characteristics of the system components, system reliability can be computed using either exact methods or approximations.

The traditional mathematical theory of system reliability assumes that both the system and component behaviors are binary in nature. That is, the system and components are either completely failed or perfectly functional. When binary behavior holds, system-reliability evaluation becomes a well-developed, extensive and robust theory. Based on this

behavior, extensive research has been performed to develop methods that represent the interactions between the system and its components (Hansler, 1972; Ball and Van Slyke, 1977; Dotson and Gobien, 1979; Torrieri, 1994; Netes and Filin, 1996; Kuo *et al.*, 1999). Similarly, computational procedures have been developed to obtain an exact or approximate system reliability once the first phase has been conducted (Esary and Proschan, 1963; Ball and Provan, 1983; Fishman, 1986; Colbourn, 1988; Jin and Coit, 2003).

However, some systems are more complicated. For example the system components may be operating in a degraded state with the system still being able to provide an acceptable level of service (Natvig, 1982; Hudson and Kapur, 1983; Aven, 1985; Natvig *et al.*, 1986; Boedigheimer and Kapur, 1994). For these systems, the theory of multistate reliability can offer modeling and computational procedures that better represent and account for system functioning.

Multistate reliability theory copes with the problem of evaluating system reliability when the components and the system operate in any of several intermediate states (Barlow and Wu, 1978; El-Newehi *et al.*, 1978; Ross, 1979; Hansler, 1972; Block and Savits, 1982). The last decade has seen the development of methods to evaluate specific multistate system architectures (Levitin *et al.*, 1998; Ramirez-Marquez and Coit, 2004). The evaluation of system reliability for

---

\*Corresponding author

more complex designs either relies on enumerative techniques (Patra and Misra, 1993, 1996; Billinton and Zhang, 2000) or considers a particular multistate behavior of the system components (Lin, 2001, 2002; Ramirez-Marquez and Coit, 2003; Yeh, 2004).

This paper is focused on the development of a method for Multistate two-Terminal Reliability (M2TR) evaluation. When the arcs and nodes of a network are considered as being multistate in nature then binary evaluation methods (Hansler, 1972; Ball and Van Slyke, 1977; Dotson and Gobien, 1979; Torrieri, 1994; Netes and Filin, 1996; Kuo *et al.*, 1999) can only provide an approximation to the actual system reliability. In the multistate case, two-terminal reliability at demand level  $d$  (M2TR $_d$ ) is defined as the probability that the system capacity, generated by the multistate arcs, is greater than or equal to a demand of  $d$  units.

This paper provides an analysis technique for the M2TR problem that is based on the extension of binary concepts to the modeling of the interactions between the multistate behavior at the component and system levels. Moreover, based on this interaction, an extension of a computational method for the approximation of the actual numerical multistate system reliability is presented.

The approach presented in this study evaluates system reliability by determining Multistate Minimal Path Vectors (MMPVs); the multistate equivalent of minimal path sets in 2TR. Once the MMPVs are obtained then one approach to obtain the M2TR $_d$  is the classical inclusion/exclusion formula (Lin, 2001, 2002; Ramirez-Marquez and Coit, 2003; Yeh, 2004). For some cases, this may be a simple and practical approach. However, as the size of the network or the number of component states increases, using inclusion/exclusion may become computationally inefficient.

Actual systems in which the M2TR $_d$  is a critical reliability metric tend to be complex systems. Reliability computation may be too costly to obtain, in terms of computational efficiency or accuracy, through traditional techniques. The algorithm presented in this paper is complemented by a Monte Carlo (MC) simulation approach, that is an improved version of the approach proposed in Ramirez-Marquez and Coit (2005a) that is able to effectively approximate the M2TR $_d$  value for relatively large systems. This MC approach is based on the identification of all the MMPVs and consists in generating component states for each component, comparing them against each MMPV and deciding if the capacity of the network satisfies the required demand.

The remainder of the paper is organized as follows. Section 2 presents an introduction to the analysis and computation of the M2TR along with current methods to solve this problem. Section 3 discusses and illustrates the proposed algorithm, and in Section 4, the approach is applied to three different multistate networks. Section 5 contains a discussion on multistate numerical reliability computations based on the algorithm output. Finally, Section 6 presents conclusions.

## 2. M2TR computation

The major focus of this research is to generalize the binary reliability theory concept of path sets to multistate reliability and present analysis methods for their computation. Elsayed (1996) defines a binary path set as a set of components that guarantees system success. Since in the multistate case system reliability is a hybrid of connectivity and capacity, the binary definition of a path set must be modified. The multistate version of path sets should consider, not only components that guarantee  $s$ - $t$  connectivity success between source node  $s$  and sink node  $t$ , but also the capacity levels at which these components guarantee  $s$ - $t$  capacity success (Hudson and Kapur, 1983; Natvig *et al.*, 1986). For the multistate case, the concept of path sets will be replaced by the notion of multistate path vectors. These vectors describe the current state of each of the system components that guarantee system success.

### 2.1. Problem description and definitions

Let  $G = (N, A)$  represent a stochastic capacitated network with known demand  $d$  between known source node  $s$  and sink node  $t$ .  $N$  represents the set of nodes and  $A = \{i | 1 \leq i \leq m\}$  represents the set of arcs. The current state (capacity) of arc  $i$  is defined by  $x_i$  which takes values from vector  $\mathbf{b}_i$ , represented by  $\mathbf{b}_i = (b_{i1} = 0, b_{i2}, \dots, b_{i\omega_i} = M_i)$  where  $b_{ij} \in \mathbf{Z}^+$ ,  $M_i$  equals the maximum capacity of arc  $i$  and  $\omega_i$  equals its number of states. That is, vector  $\mathbf{b}_i$  represents the range of states (capacities) associated with arc  $i$ . The vector  $\mathbf{p}_i = (p_{i1}, p_{i2}, \dots, p_{i\omega_i})$  represents the probability associated with each of the values taken by  $x_i$  (i.e.,  $p_{ij} = P(x_i = b_{ij})$ ). The system state vector  $\mathbf{x} = (x_1, x_2, \dots, x_m)$  denotes the state of all the arcs of the network. Function  $\varphi: \mathbf{Z}^m \rightarrow \mathbf{Z}^+$ , maps the system state vector into a system state. That is,  $\varphi(\mathbf{x})$  is the network capacity between  $s$  and  $t$  under system state vector  $\mathbf{x}$ . M2TR $_d$  denotes the probability that a demand of  $d$  units can be supplied from source node to sink node through the multistate arcs:  $\text{M2TR}_d = P\{\varphi(\mathbf{x}) \geq d\}$ .

We make the following assumptions.

1. The component states are mutually stochastically independent.
2. The structure function  $\varphi(\mathbf{x})$  is coherent. That is, improving component performance cannot cause the network to become less reliable.
3. The component states and their associated probabilities are known.

#### 2.1.1. Vector properties

- A vector  $\mathbf{y}$  is said to be less than  $\mathbf{x}$ , or  $\mathbf{y} < \mathbf{x}$ , iff for every  $y_i$  (the  $i$ th entry of vector  $\mathbf{y}$ ) and  $x_i$  (the  $i$ th entry of vector  $\mathbf{x}$ ),  $y_i \leq x_i$  and for some  $y_k$  (the  $k$ th entry of vector  $\mathbf{y}$ ), and  $x_k$  (the  $k$ th entry of vector  $\mathbf{x}$ ),  $y_k < x_k$ .
- A vector  $\mathbf{y}$  is said to be greater than  $\mathbf{x}$ , or  $\mathbf{y} > \mathbf{x}$ , iff for every  $y_i$  (the  $i$ th entry of vector  $\mathbf{y}$ ) and  $x_i$  (the  $i$ th entry of

vector  $\mathbf{x}$ )  $y_i \geq x_i$  and for some  $y_k$  (the  $k$ th entry of vector  $\mathbf{y}$ ) and  $x_k$  (the  $k$ th entry of vector  $\mathbf{x}$ ),  $y_k > x_k$ .

- A vector  $\mathbf{y}$  is said to be equal to  $\mathbf{x}$ , or  $\mathbf{y} = \mathbf{x}$ , iff for every  $y_i$  (the  $i$ th entry of vector  $\mathbf{y}$ ) and  $x_i$  (the  $i$ th entry of vector  $\mathbf{x}$ ),  $y_i = x_i$ .
- A vector  $\mathbf{x}$  is said to be a MMPV at level  $d$  if  $\varphi(\mathbf{x}) \geq d$  and for every other  $\mathbf{y} < \mathbf{x}$ ,  $\varphi(\mathbf{y}) < d$ .
- A vector  $\mathbf{x}$  is said to be a Multistate Minimal Cut Vector (MMCV) at level  $d$  if  $\varphi(\mathbf{x}) < d$  and for every other  $\mathbf{y} > \mathbf{x}$ ,  $\varphi(\mathbf{y}) \geq d$ .

### 2.1.2. Weakly homogeneous components

Let  $\mathbf{b}_h = (b_{h1}, b_{h2}, \dots, b_{hMh})$  and  $\mathbf{b}_k = (b_{k1}, b_{k2}, \dots, b_{kMk})$  be the state space vectors of components  $a_h$  and  $a_k$  respectively. Components  $a_h$  and  $a_k$  are called weakly homogeneous if the first  $l_k$  component states are equal. That is, components can have a different number of states yet for components,  $a_h$  and  $a_k$  with  $|\mathbf{b}_h| = l_h$  and  $|\mathbf{b}_k| = l_k$  and  $l_h > l_k$ , where  $|\mathbf{b}_h|$  and  $|\mathbf{b}_k|$  represent the cardinality of the state vectors  $\mathbf{b}_h$  and  $\mathbf{b}_k$  of components  $a_h$  and  $a_k$ , the first  $l_k$  component states must be equal. Components that are not initially weakly homogeneous can often be transformed into weakly homogeneous components by introducing new component states with a zero associated probability.

### 2.2. On multistate minimal vectors

The system success (failure) at a demand level  $d$  can be completely characterized by the set containing all the minimal path (cut) vectors. To clarify these concepts, consider the series system depicted in Fig. 1. This system has  $m$  arcs and the system capacity must be at least  $d$  for the system to function properly. All vectors that guarantee system success (failure) can be obtained based on the definition of minimal path (cut) vectors at level  $d$  MMPVs (MMCVs).

For the system to fulfill demand  $d$ , each of the system components must have at least a capacity of  $d$ . Assuming that the nominal capacity of each of the components is greater than or equal to  $d$ , then a single MMPV,  $\text{MMPV}_1$ , can be obtained:  $\text{MMPV}_1 = (x_1, x_2, \dots, x_m)$  where  $x_i$  represents the current capacity of arc  $a_i$  and the value of  $x_i$  is obtained as follows:  $x_i = \min_j \{b_{ij} | b_{ij} \geq d\} \forall i = 1, \dots, m$ .

On the other hand, system failure is guaranteed when the capacity of any component is strictly less than  $d$ . Thus, there are  $m$  distinct MMCVs,  $\text{MMCV}_i, i = 1, \dots, m$ , of the form:  $\text{MMCV}_i = (M_1, M_2, \dots, x_i, M_{i+1}, \dots, M_m)$  where  $x_i$  represents the capacity of arc  $a_i$  and the value of  $x_i$  is obtained as follows:  $x_i = \max_j \{b_{ij} | b_{ij} < d\} \forall i = 1, \dots, m$ .



Fig. 1. Series system configuration.

### 2.3. Reliability computation

Based on these vectors the  $\text{M2TR}_d$  can be computed through the inclusion/exclusion formula. For binary reliability this formula is defined as:

$$R = 1 - P\left(\bigcup_{h=1}^l C_h\right) = 1 - \sum_{h=1}^l P(C_h) + \sum_{h<k} P(C_h \cap C_k) - \dots (-1)^l P(C_1 \cap C_2 \cap \dots \cap C_l),$$

where  $l$  is the number of minimal cut sets, and  $C_h$  is the minimal cut set  $h$ . Equivalently:

$$R = P\left(\bigcup_{h=1}^t P_h\right) = \sum_{h=1}^t P(P_h) - \sum_{h<k} P(P_h \cap P_k) + \dots (-1)^t P(P_1 \cap P_2 \cap \dots \cap P_t),$$

where  $t$  is the number of minimal path sets, and  $P_h$  is the minimal path set  $h$ .

This formula has to be extended to account for the new vector structure of the minimal sets. For the multistate case the  $\text{M2TR}_d$  can be obtained with the following modification of the inclusion/exclusion formula:

$$\begin{aligned} \text{M2TR}_d &= 1 - \sum_{h=1}^L P(\mathbf{x} \leq \text{MMCV}_h) \\ &+ \sum_{h<k} P(\mathbf{x} \leq \text{MMCV}_h \wedge \mathbf{x} \leq \text{MMCV}_k) \\ &- \dots (-1)^L P(\mathbf{x} \leq \text{MMCV}_1 \wedge \mathbf{x} \leq \text{MMCV}_2 \\ &\wedge \dots \wedge \mathbf{x} \leq \text{MMCV}_L) \end{aligned}$$

where  $L$  is the number of MMCVs and  $\text{MMCV}_h$  is the  $h$ th MMCV such that:

$$P(\mathbf{x} \leq \text{MMCV}_h) = \prod_{i=1}^m P(x_i \leq y_i) \quad \forall x_i \in \mathbf{x}$$

and every  $y_i \in \text{MMCV}_h$ .

In order to compute the remaining terms in the formula: we must have that:

$$\begin{aligned} P(\mathbf{x} \leq \text{MMCV}_h \wedge \mathbf{x} \leq \text{MMCV}_k) \\ = \prod_{i=1}^m P(x_i \leq \min_i \{y_i, z_i\}) \quad \forall x_i \in \mathbf{x}, \end{aligned}$$

and every  $y_i \in \text{MMCV}_h$  and  $z_i \in \text{MMCV}_k$ .

Equivalently:

$$\begin{aligned} \text{M2TR}_d &= \sum_{h=1}^T P(\mathbf{x} \geq \text{MMPV}_h) - \sum_{h<k} P(\mathbf{x} \geq \text{MMPV}_h \wedge \\ &\mathbf{x} \geq \text{MMPV}_k) + \dots (-1)^T P(\mathbf{x} \geq \text{MMPV}_1 \wedge \\ &\mathbf{x} \geq \text{MMPV}_2 \wedge \dots \wedge \mathbf{x} \geq \text{MMPV}_T), \end{aligned}$$

where  $T$  is the number of MMPVs and  $\text{MMPV}_h$  is the  $h$ th MMPV such that:

$$P(\mathbf{x} \geq \text{MMPV}_h) = \prod_{i=1}^m P(x_i \geq y_i) \quad \forall x_i \in \mathbf{x},$$

and every  $y_i \in \text{MMPV}_h$ . In order to compute the remaining terms in the formula: we must have that:

$$P(\mathbf{x} \geq \text{MMPV}_h \wedge \mathbf{x} \geq \text{MMPV}_k) = \prod_{i=1}^m P(x_i \geq \max_i \{y_i, z_i\}) \\ \forall x_i \in \mathbf{x}$$

and every  $y_i \in \text{MMPV}_h$  and  $z_i \in \text{MMPV}_k$ .

The reliability of a system that follows the two-terminal rationale can be obtained based on the transformed inclusion/exclusion formulas. The problem reduces to finding either the MMCV or MMPV at the required demand level.

#### 2.4. M2TR Approaches

For systems where binary-state analysis is insufficient, an incorrect reliability assessment can lead to faulty decision-making with regard to network performance. Unnecessary expenditures, incorrect maintenance scheduling and a reduction in safety standards are problems that potentially can be related to unsatisfactory reliability assessments. Thus, there is a real need to incorporate a more realistic view of system performance concerning the multistate behavior of systems.

Recently, multistate network reliability modeling has attracted the attention of network reliability researchers. The first approach to solve this problem was to use enumerative methods as in the papers of Hudson and Kapur (1983), Aven (1985), Natvig *et al.* (1986), and Boedigheimer and Kapur (1994). These methods relate all possible combinations of the component states to a system state. Thus, it is understandable that enumerative procedures can be applied only to relatively small systems.

The method proposed by Patra and Misra (1993, 1996) is an enumerative procedure that uses binary cut sets to develop multistate system state vectors that potentially satisfy the system demand. Each of these vectors is analyzed by filtering out all these infeasible vectors that do not fulfill the required demand from source to sink. This method does not provide either multistate paths or cut vectors. In fact, in some cases this approach may lead to a complete enumeration procedure.

Lin (2001) and Ramirez-Marquez and Coit (2003) have developed approaches to compute the exact M2TR value using a multistate version of minimal path sets. The algorithm presented by Lin (2001) considered weakly homogeneous components. Thus, for systems in which the components are heterogeneous, the method must be improved. A network reduction technique is described in Ramirez-Marquez and Coit (2003) that is able to obtain all possible MMPVs. If binary components are considered than the method reduces to the approach developed by Kuo *et al.* (1999). These approaches have only been applied to relatively small networks.

Lin (2002) and Yeh (2004) presented the first approaches to identify the multistate version of minimal binary cut sets

for the M2TR problem. The MMCVs are computed in a similar form. For each binary cut, they enumerate all different state combinations of the components in the cut to obtain the multistate levels that guarantee a MMCV. These methods consider weakly homogeneous components with the added constraint that the capacity of arc  $i$  is an integer-valued random variable. This research represented an important contribution to the problem solution. However, the algorithms cannot be applied to problems of the type presented by Patra and Misra (1993, 1996).

The algorithm presented by Ramirez-Marquez *et al.* (2004) reduces the computational burden inherent in previous approaches (Lin, 2002; Yeh, 2004). The motivation behind the algorithm is two-fold. Firstly, a binary minimal cut vector is a cut vector in the multistate case, although it may not be minimal. That is, the level,  $x_i$ , at which the arcs belonging to the binary cut operation, may not be minimal when considering multistate behavior. Thus, the problem in the multistate case reduces to finding the minimal state levels of the components in the binary cut. The algorithms previously developed use this property to construct the MMCVs. Secondly, information sharing between the cuts can significantly reduce the number of enumerations needed to obtain the MMCVs. Based on the binary minimal cut sets, the algorithms of Lin (2002) and Yeh (2004) use exhaustive enumeration to find the new component states of a given binary cut that entail system failure in the multistate case. The approach of Ramirez-Marquez *et al.* (2004), however, identifies two basic cuts, called parent cuts, which propagate information (component states) to a specified number of binary cuts called offspring cuts.

Levitin *et al.* (1998) use a procedure based on the concept of a universal generating function to compute M2TR<sub>d</sub>. This technique has proven to be a valuable and efficient tool for relatively complex systems. It requires relatively small computational resources to evaluate multistate reliability indices. A detailed description of the universal generating function is presented in Levitin and Lisnianski (2003).

Finally, it is important to note that most methods to compute M2TR are based on providing MMCVs. Since the number of potential MMCVs generated by these methods is a function of the network binary cut sets, these approaches are usually preferable for use in cases in which the binary minimal cut sets can be readily obtained and the total number of these sets is of a manageable size. However, for some networks a multistate-path-based approach is more attractive because:

1. The MMPVs can be used as a potential indicator of the different scenarios in which the network is successful. Hence, finding these network state vectors provides a good method to study and prioritize these components that should be improved or backed-up to ensure that the network sustains high levels of reliability (Aven and Østebø, 1986; Ramirez-Marquez and Coit, 2005b).

2. The number of state enumerations and vector analyses to obtain the MMPVs can be significantly reduced as compared with MMCV methods. That is, the number of enumerations to obtain the MMCVs is closely related to the number of binary minimal cut sets. As the network size increases the problem of identifying these sets becomes more difficult for larger and more complex systems (Fotuhi-Firuzabad *et al.*, 2004; Kuo *et al.*, 1999).
3. The total number of MMPVs may be made more manageable than the number of MMCVs, thus contributing to a faster M2TR exact computation (Ramirez-Marquez and Coit, 2005a).

### 3. M2TR through a MMPV approach

#### 3.1. Discussion of the approach

The algorithm presented in this paper is a fully multistate approach. Current M2TR computation methods assume that the binary cut sets of network  $G$  are known *a priori*. Thus, when no information about these sets is available then the algorithms are not applicable. Lin (2001) and Lin *et al.* (1995) have developed methods to compute the MMPVs by developing and solving sets of inequalities imposed by the structure of the network. These approaches are problem specific, and for each particular network, a set of inequalities must be developed and solved. Furthermore, they depend on knowing the binary minimal path sets *a priori*.

Unlike the MMCV case (Lin, 2002; Yeh, 2004), a binary minimal path vector is not necessarily a multistate path vector. The MMPV computation problem is complicated because different combinations of arc levels must be analyzed in order to obtain the MMPVs and ultimately M2TR<sub>*d*</sub>. The inequalities developed by use of binary path sets and system requirements in Lin (2001) and Lin *et al.* (1995) reduce the search space in which to obtain the MMPVs. The approach presented in this section can be regarded as a network reduction or transformation method. Once the network is reduced, information sharing among the removed and successor arcs reduces the number of vector analyses needed to obtain the MMPVs. This reduction and information-sharing step is iteratively performed until all network arcs have been examined.

#### 3.2. Algorithm description

The rationale behind the algorithm mimics the MMCV approach of Ramirez-Marquez *et al.* (2004) in the sense that a select number of network arcs share information between themselves to reduce the number of state enumerations needed to obtain all MMPVs. The information sharing mechanism allows for an effective computation of the network's MMPVs without the need to rely on the binary minimal path sets.

The initial step of the algorithm requires that a successor matrix be computed along with a primary set of components. Initially, the "primary set" is constructed by considering the set containing all arcs that originate at the source node. The second step identifies component states ( $x_i$ ) that guarantee that the components in the primary set satisfy the demand requirement used to set up the information sharing process. Step 3 identifies successors of the components in the primary set via the successor matrix and passes information about the component states ( $x_i$ ) that should be analyzed. Once these successors have been exhausted, the primary set is updated and the process is repeated. The updated primary set is constructed by considering the set containing all the successors analyzed in the previous run.

At each process run (primary set update) potential MMPVs are updated. When no successor is left to analyze, i.e., the primary set can no longer be updated, functions  $\mu$  and  $\delta$  check for vector feasibility and MMPV condition, respectively. Finally, M2TR<sub>*d*</sub> can be computed using inclusion/exclusion formula or approximated through a MC simulation approach similar to Ramirez-Marquez and Coit (2005a).

It should be noted that this algorithm significantly differs from current approaches reported in the literature in the following respects: i) it does not depend on *a priori* knowledge of minimal path or cut sets; and (ii) although it uses an information sharing technique initially used in Ramirez-Marquez *et al.* (2004) the application of the technique in the proposed algorithm is entirely innovative because of the way the arcs are selected to inherit information.

#### Algorithm pseudo-code

Initialize:

$$x_i = 0 \quad \forall \quad i = 1, \dots, m, \quad P_1 = \Gamma = T = \Omega = \Psi = T' = \emptyset, \quad r = 0, \quad d = \text{source-sink requirement.}$$

Step 1. Successor matrix and primary set

1.1. Compute successor matrix  $U$ .

$$u_{ij} = \begin{cases} 1 & \text{arc } j \text{ is a successor of arc } i \\ 0 & \text{otherwise} \end{cases}$$

1.2. Define primary set  $P_1$ :

$$P_1 = \{a_i | a_i \text{ originates at the source node}\} \\ T' \rightarrow P_1$$

Step 2. Initial potential MMPVs

Identify all combinations of states  $b_{ij}$  such that:

$$\sum_{a_i \in T'} b_{ij} = d$$

For each combination, create vectors  $\mathbf{x}$  via:

$$x_i = \begin{cases} x_i + b_{ij} & \text{if } a_i \in T', \\ x_i & \text{otherwise.} \end{cases}$$

Update  $\Omega \rightarrow \Omega \cup \{\mathbf{x}\}$

Step 3. Information sharing

3.1. If  $P_1 \neq \emptyset$

$$r \rightarrow \min \{k | a_k \in P_1\}$$

$$T \rightarrow a_r$$

$$P_1 \rightarrow (T \cap P_1)^c$$

$$T' = \{a_k | u_{kr} = 1\}$$

3.2. If  $T' = \emptyset$

Return to Step 3.1

If  $a_r \in \Psi$

Return to Step 3.1

Else

$$\Psi \rightarrow T \cup \Psi$$

$$\Gamma \rightarrow \Gamma \cup T'$$

$$\Omega' \rightarrow \Omega; \Omega \rightarrow \emptyset$$

For each  $x_i$  in a potential MMPV

$$\mathbf{x} \in \Omega'$$

Identify all combinations of states

$b_{ij}$  such that:

$$\sum_{a_i \in T'} b_{ij} = x_r$$

For each combination, create vectors  $\mathbf{x}$  via:

$$x_i = \begin{cases} x_i + b_{ij} & \text{if } a_i \in T' \\ x_i & \text{otherwise} \end{cases}$$

Update  $\Omega \rightarrow \Omega \cup \{\mathbf{x}\}$

Return to Step 3.1

If  $\Gamma = \emptyset$

$$\Omega' \rightarrow \Omega$$

While  $\Omega' \neq \emptyset$

Take  $\mathbf{x} \in \Omega'$ . Go to Step 4

$$\Omega \rightarrow (\Omega \cap \Theta)^c$$

For every  $\mathbf{x} \in \Omega$  and  $\mathbf{y} \in \Omega$ ,  $\mathbf{x} \neq \mathbf{y}$ . Go to Step 5

STOP: The multistate minimal path vectors are contained in set  $\Theta$

Else

$$P_1 \rightarrow \Gamma$$

$$\Gamma \rightarrow \emptyset$$

Go to Step 3.1

Step 4. Function  $\mu(\mathbf{x})$

$$\Omega' \rightarrow (\Omega' \cap \{\mathbf{x}\})^c$$

For  $i = 1, \dots, m$

If  $x_i > M_i$ ,  $x_i \in \mathbf{x}$

$$\Theta \rightarrow \Theta \cup \mathbf{x}$$

Else

Step 5. Function  $\delta(\mathbf{x}, \mathbf{y})$

If  $\mathbf{x} \geq \mathbf{y}$

$$\Theta \rightarrow \Theta \cup \mathbf{x}$$

Return  $\Theta$

Else  $\Theta \rightarrow \Theta$

Return  $\Theta$

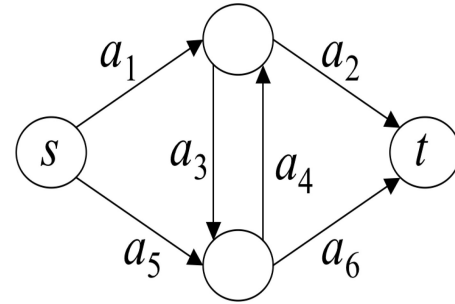


Fig. 2. The multistate network due to Lin (2001).

4. Examples and comparison with commonly used methods

In this section, several examples are provided to demonstrate the algorithm. Example 1 is taken from Lin (2001) and each step of the algorithm and the computations needed are presented at each of the iterations to obtain the MMPVs. Example 2 considers the same data as example 1 with a variation of system demand to provide differing MMPVs. Complete enumeration requires the computation and capacity analysis of 2304 state vectors for each problem. Lin (2001) and Lin *et al.* (1995) have the added complexity of obtaining minimal path sets and creating and solving problem-specific inequalities to obtain the MMPVs. This is the first time a fully multistate approach has been used to solve this type of network. That is, the proposed algorithm does not depend on the *a priori* computation of binary minimal path sets. Finally, example 3 considers a multistate system initially analyzed by Levitin *et al.* (2003). This last example is used to show how the proposed approach can be used to obtain MMPVs in systems where components have heterogeneous states.

4.1. Example 1

Figure 2 presents the network analyzed by Lin (2001). It is desired to obtain all the MMPVs that guarantee that the system capacity from the source to sink node is greater than or equal to five units. Table 1 presents the states of each of the network arcs. The following figure and sequence of

Table 1. The states of the network arcs.

$a_i$	States			
	$b_{i1}$	$b_{i2}$	$b_{i3}$	$b_{i4}$
1	0	1	2	
2	0	1	2	3
3	0	1	2	
4	0	1	2	3
5	0	1	2	3
6	0	1	2	3

Successor Matrix						
$u_{ij}$	$a_1$	$a_2$	$a_3$	$a_4$	$a_5$	$a_6$
$a_1$	0	1	1	0	0	0
$a_2$	0	0	0	0	0	0
$a_3$	0	0	0	0	0	1
$a_4$	0	1	0	0	0	0
$a_5$	0	0	0	1	0	1
$a_6$	0	0	0	0	0	0

Fig. 3a. The successor matrix.

tables illustrates how the potential MMPVs are constructed and updated based on the primary set and on the inherited values.

The initial step requires input of the successor matrix, shown in Fig. 3a, and the primary set. Note that in the construction of the predecessor matrix arcs  $a_3$  and  $a_4$  are not considered to be successors of one another. See the note on cycle handling in Section 4.1.1. In this example the primary set is  $P_1 = \{a_1, a_5\}$ . From this primary set, shown in Table 2, the information sharing approach will be initiated by developing vectors as dictated in Step 2 of the algorithm.

(Step 3.1.) The values of the sets are updated in the following form:  $T \rightarrow \{a_1\}$ ,  $T' \rightarrow \{a_2, a_3\}$ ,  $P_1 \rightarrow \{a_5\}$ ,  $\Gamma \rightarrow \{a_2, a_3\}$ . (Step 3.2) The potential MMPV are listed in Table 3.

(Step 3.1.) Sets are updated:  $T \rightarrow \{a_5\}$ ,  $T' \rightarrow \{a_4, a_6\}$ ,  $P_1 \rightarrow \emptyset$ ,  $\Gamma \rightarrow \{a_2, a_3, a_4, a_6\}$ . (Step 3.2) This gives the potential MMPVs listed in Table 4.

The initial primary set has been analyzed and must be updated. This process is repeated until the primary set can no longer be updated. The remaining steps follow.

(Step 3.1.) Sets are updated:  $P_1 \rightarrow \{a_2, a_3, a_4, a_6\}$ ,  $T \rightarrow \{a_2\}$ ,  $P_1 \rightarrow \{a_3, a_4, a_6\}$ ,  $T' \rightarrow \emptyset$ ,  $\Gamma \rightarrow \emptyset$ . No potential MMPV is updated since the successor set is empty. A new updating process is performed:

(Step 3.1.)  $T \rightarrow \{a_3\}$ ,  $P_1 \rightarrow \{a_4, a_6\}$ ,  $T' \rightarrow \{a_6\}$ ,  $\Gamma \rightarrow \{a_6\}$ . (Step 3.2.) The potential MMPVs in this case are listed in Table 5.

(Step 3.1.) Now  $T \rightarrow \{a_4\}$ ,  $P_1 \rightarrow \{a_6\}$ ,  $T' \rightarrow \{a_2\}$ ,  $\Gamma \rightarrow \{a_2, a_6\}$ . (Step 3.2.) The potential MMPVs listed in Table 6 are computed.

Sets are updated:  $T \rightarrow \{a_6\}$ ,  $P_1 \rightarrow \emptyset$ ,  $T' \rightarrow \emptyset$ ,  $\Gamma \rightarrow \{a_2, a_6\}$ . No potential MMPV can be updated since the set containing successors is empty. Moreover, the primary set is empty thus a new process must be started.

Table 2. The primary set

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
1	2	0	0	0	3	0

Table 3. The potential MMPVs

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
1	2	0	2	0	3	0
2	2	1	1	0	3	0
3	2	2	0	0	3	0

Table 4. The MMPVs after the second updating process

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
1	2	0	2	3	3	0
2	2	0	2	2	3	1
3	2	0	2	1	3	2
4	2	0	2	0	3	3
5	2	1	1	3	3	0
6	2	1	1	2	3	1
7	2	1	1	1	3	2
8	2	1	1	0	3	3
9	2	2	0	3	3	0
10	2	2	0	2	3	1
11	2	2	0	1	3	2
12	2	2	0	0	3	3

Table 5. The MMPVs after the fourth updating process

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
1	2	0	2	3	3	2
2	2	0	2	2	3	3
3	2	0	2	1	3	4
4	2	0	2	0	3	5
5	2	1	1	3	3	1
6	2	1	1	2	3	2
7	2	1	1	1	3	3
8	2	1	1	0	3	4
9	2	2	0	3	3	0
10	2	2	0	2	3	1
11	2	2	0	1	3	2
12	2	2	0	0	3	3

Table 6. The MMPVs after the fifth updating process

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
1	2	3	2	3	3	2
2	2	2	2	2	3	3
3	2	1	2	1	3	4
4	2	0	2	0	3	5
5	2	4	1	3	3	1
6	2	3	1	2	3	2
7	2	2	1	1	3	3
8	2	1	1	0	3	4
9	2	5	0	3	3	0
10	2	4	0	2	3	1
11	2	3	0	1	3	2
12	2	2	0	0	3	3

**Table 7.** The inadequate vectors

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
3	2	1	2	1	3	4
4	2	0	2	0	3	5
5	2	4	1	3	3	1
8	2	1	1	0	3	4
9	2	5	0	3	3	0
10	2	4	0	2	3	1

$P_1 \rightarrow \{a_2, a_6\}, T \rightarrow \{a_2\}, P_1 \rightarrow \{a_6\}, T' \rightarrow \emptyset, \Gamma \rightarrow \emptyset$ . No potential MMPV can be updated since the set containing successors is empty.  $T \rightarrow \{a_6\}, P_1 \rightarrow \emptyset, T' \rightarrow \emptyset, \Gamma \rightarrow \emptyset$ . The process has been exhausted since the updated primary set equals the null set. The next step applies the adequacy function  $\mu$ . This function identifies the vectors listed in Table 7 as being inadequate.

For the remaining vectors function  $\delta$  checks the condition of the MMPVs, this function yields the MMPVs listed in Table 8.

4.1.1. *Cycle handling*

One of the user inputs into the algorithm is the successor matrix. This matrix contains information on the network structure. If cyclic networks occur then the successor matrix has to be constructed with special care to ensure that the algorithm does not generate infeasible MMPVs. As an example, consider the successor matrix associated with the network depicted in Fig. 2. According to the definition of  $u_{ij}, u_{34}$  and  $u_{43}$  should have a value equal to one. However, if these values were coded into the successor matrix the algorithm would cycle between these arcs, create infeasible values for both  $x_3$  and  $x_4$  and impose these values on the successors of these arcs. In summary, the network structure and associated connectivity protocol must be well understood before using the successor matrix.

4.2. *Example 2*

The second example considers the same network and data given for example 1. However, the demand requirement is reduced to four units. Table 9 presents the potential MMPVs obtained using the proposed algorithm. These vectors are checked first with function  $\mu$  to test for adequacy. The adequate vectors, after application of function  $\mu$ , are presented in Table 10. Finally, Table 11 presents the final vectors after the use of function  $\delta$  to check for the MMPV condition.

**Table 8.** The MMPVs yielded by the  $\delta$  function

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
11	2	3	0	1	3	2
12	2	2	0	0	3	3

**Table 9.** Potential MMPVs for example 2

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
1	3	4	0	1	1	0
2	3	3	0	0	1	1
3	3	3	1	1	1	1
4	3	2	1	0	1	2
5	3	2	2	1	1	2
6	3	1	2	0	1	3
7	3	1	3	1	1	3
8	3	0	3	0	1	4
9	2	4	0	2	2	0
10	2	3	0	1	2	1
11	2	2	0	0	2	2
12	2	3	1	2	2	1
13	2	2	1	1	2	2
14	2	1	1	0	2	3
15	2	2	2	2	2	2
16	2	1	2	1	2	3
17	2	0	2	0	2	4
18	1	4	0	3	3	0
19	1	3	0	2	3	1
20	1	2	0	1	3	2
21	1	1	0	0	3	3
22	1	3	1	3	3	1
23	1	2	1	2	3	2
24	1	1	1	1	3	3
25	1	0	1	0	3	4

**Table 10.** The adequate vectors for example 2

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
10	2	3	0	1	2	1
11	2	2	0	0	2	2
12	2	3	1	2	2	1
13	2	2	1	1	2	2
14	2	1	1	0	2	3
15	2	2	2	2	2	2
16	2	1	2	1	2	3
19	1	3	0	2	3	1
20	1	2	0	1	3	2
21	1	1	0	0	3	3
22	1	3	1	3	3	1
23	1	2	1	2	3	2
24	1	1	1	1	3	3

**Table 11.** The MMPVs for example 2

$x_i$	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$
10	2	3	0	1	2	1
11	2	2	0	0	2	2
14	2	1	1	0	2	3
19	1	3	0	2	3	1
20	1	2	0	1	3	2
21	1	1	0	0	3	3

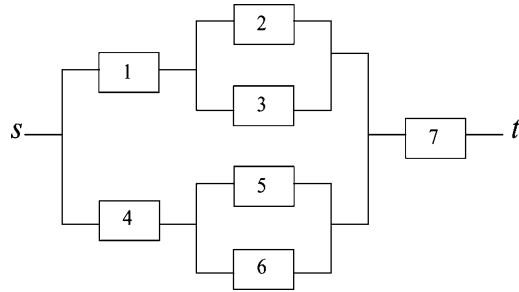


Fig. 3b. The system considered by Levitin *et al.* (2003)

4.3. Example 3

The system depicted in Fig. 3b with data given in Table 12 was first analyzed using the universal generating function method by Levitin *et al.* (2003). The system must supply five units from the source to the sink nodes. Since the component states are heterogeneous no technique other than our method can be applied. Our proposed algorithm can provide the MMPVs by transforming the heterogeneous components into homogeneous components in the form of “dummy” states. Table 13 presents the transformed component data that allows for application of our proposed method. The vectors obtained by using our algorithm are transformed into system MMPVs by the values of the component’s “dummy” states for its next highest original state. Table 14 shows the list of heterogeneous MMPVs.

5. Computation of the system reliability

The previous examples illustrated how the algorithm can effectively obtain MMPVs in networks with homogeneous and heterogeneous component states. If the component state probabilities are known then the vectors obtained through the algorithms can be used to compute or approximate the actual M2TR<sub>d</sub> value.

It is important to note that the approaches of Lin *et al.* (1995), Lin (2001, 2002), Ramirez-Marquez and Coit (2003), Ramirez-Marquez *et al.* (2004) and Yeh (2004) to the analysis of M2TR<sub>d</sub> assume that once the MMCVs or MMPVs are obtained then the reliability computation

Table 12. The system data of Levitin *et al.* (2003)

$a_i$	$b_{i1}$	$b_{i2}$	$b_{i3}$	$b_{i4}$	$b_{i5}$
1	0	1	3	5	7
2	0	2	4		
3	0	2	4		
4	0	2	6	8	
5	0	2	4		
6	0	2	4		
7	0	6	10	14	18

Table 13. The transformed system data

$a_i$	$b_{i1}$	$b_{i2}$	$b_{i3}$	$b_{i4}$	$b_{i5}$	$b_{i6}$	$b_{i7}$	$b_{i8}$	$b_{i9}$	$b_{i10}$
1	0	1	2	3	4	5	7			
2	0	1	2	3	4					
3	0	1	2	3	4					
4	0	1	2	3	4	5	6	8		
5	0	1	2	3	4					
6	0	1	2	3	4					
7	0	1	2	3	4	5	6	10	14	18

is a straightforward process using the transformed inclusion/exclusion formula for multistate reliability. However, as the size of the network or the number of component states increases, the inclusion/exclusion approach becomes impractical because the number of minimal vectors needing to be analyzed can increase considerably.

Recently, Ramirez-Marquez and Coit (2005a) developed a MC simulation approach to the analysis of M2TR based on MMCVs. A modification of this procedure can be used to approximate M2TR values based on MMPVs. The modification is as follows. At each run of the simulation, a system state vector is generated based on the state occupancy probabilities dictated by vector  $\mathbf{p}_i$ . This vector is then compared against the set of MMPVs to decide if it corresponds to a system success  $\varphi(\mathbf{x}) \geq d$ . If the simulated vector is greater than or equal to even one of the MMPVs, then it represents a system success. At each run, the indicator variable  $K$  (system success count) is updated.

This modification not only allows for the computation of M2TR values based on MMPVs, but also allows the modified MC method to be used to verify that the obtained vectors are in fact the MMPVs of the system. If both

Table 14. The heterogeneous MMPVs

MMPV	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$	$x_7$
1	5	4	2	0	0	0	6
2	5	2	4	0	0	0	6
3	3	4	0	2	2	0	6
4	3	4	0	2	0	2	6
5	3	2	2	2	2	0	6
6	3	2	2	2	0	2	6
7	3	0	4	2	2	0	6
8	3	0	4	2	0	2	6
9	3	2	0	3	4	0	6
10	1	2	0	6	4	0	6
11	1	2	0	6	2	2	6
12	1	2	0	6	0	4	6
13	1	0	2	6	4	0	6
14	1	0	2	6	2	2	6
15	1	0	2	6	0	4	6
16	0	0	0	6	4	2	6
17	0	0	0	6	2	4	6

**Table 15.** The MMCVs obtained using the method of Ramirez-Marquez *et al.* (2004)

MMCV	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$	$x_7$
1	7	4	4	8	4	4	0
2	7	4	0	8	0	0	18
3	7	0	4	8	0	0	18
4	7	0	0	8	4	0	18
5	7	0	0	8	0	4	18
6	7	2	2	8	0	0	18
7	7	2	0	8	2	0	18
8	7	2	0	8	0	2	18
9	7	0	2	8	2	0	18
10	7	0	2	8	0	2	18
11	7	0	0	8	2	2	18
12	3	4	4	8	0	0	18
13	0	4	4	8	4	0	18
14	0	4	4	8	0	4	18
15	1	4	4	8	2	0	18
16	1	4	4	8	0	2	18
17	0	4	4	8	2	2	18
18	7	4	0	0	4	4	18
19	7	0	4	0	4	4	18
20	7	2	0	2	4	4	18
21	7	0	2	2	4	4	18
22	7	2	2	0	4	4	18
23	3	4	4	0	4	4	18
24	1	4	4	2	4	4	18

the MMCVs and MMPVs of a given network are known for some demand level then their respective MC procedure should yield the same failure or success count (provided that the same seed for the random number generator is used). Table 15 presents MMCVs obtained with the method of Ramirez-Marquez *et al.* (2004) for example 3 solved in Section 4. Table 16 presents probability data for the component states for three different test cases. Finally, Table 17 shows the number of successes and failures by considering the MMCV and MMPV simulation procedures separately (same seed and number of runs). Each procedure consists of 1000 000 runs. As shown by the results the complement of the success count equals the failure count for each of the cases.

**Table 16.** The probabilities for the component states

Case 1						Case 2						Case 3					
$a_i$	$p_{i1}$	$p_{i2}$	$p_{i3}$	$p_{i4}$	$p_{i5}$	$a_i$	$p_{i1}$	$p_{i2}$	$p_{i3}$	$p_{i4}$	$p_{i5}$	$a_i$	$p_{i1}$	$p_{i2}$	$p_{i3}$	$p_{i4}$	$p_{i5}$
1	0.003	0.003	0.003	0.001	0.99	1	0.025	0.025	0.025	0.025	0.9	1	0.025	0.025	0.025	0.025	0.9
2	0.005	0.005	0.99			2	0.05	0.05	0.9			2	0.013	0.013	0.975		
3	0.005	0.005	0.99			3	0.05	0.05	0.9			3	0.075	0.075	0.85		
4	0.003	0.003	0.004	0.99		4	0.033	0.033	0.034	0.9		4	0.003	0.003	0.003	0.99	
5	0.005	0.005	0.99			5	0.05	0.05	0.9			5	0.025	0.025	0.95		
6	0.005	0.005	0.99			6	0.05	0.05	0.9			6	0.025	0.025	0.95		
7	0.003	0.003	0.003	0.001	0.99	7	0.025	0.025	0.025	0.025	0.9	7	0.025	0.025	0.025	0.025	0.9

**Table 17.** The results for example 3

Approach	Case		
	1	2	3
MMPV*	996 931	964 969	973 167
MMCV**	3069	35 031	26 833
Runs	1000 000	1000 000	1000 000

\*Values indicate the total number of successes.  
 \*\*Values indicate the total number of failures.

## 6. Future research and conclusions

### 6.1. Future research

As previously discussed the proposed algorithm requires the user to input the successor matrix of the network. Currently, this matrix has to be manually constructed with special attention being paid to ensure that the network has no cycles so that the algorithm does not generate infeasible MMPVs or infinite loops. As part of their research into network reliability, the authors are working on an algorithm to automate the construction of successor and predecessor matrices to obtain binary minimal cut and path sets. However, a general rule for the construction of the predecessor matrix is to: (i) identify network cycles by inspection and for each cycle create a set containing the arcs associated with it; (ii) for each set identify the arc for which the removal of its successor eliminates the cycle yet does not interfere with potential *s-t* paths; and, (iii) the successors (within the cycle) of the chosen arc should not be included in the matrix. Figure 4 presents two networks that are used to illustrate this general rule in the following discussion.

- For network A, in Fig. 4 the arcs that cause cycles are completely contained within the shaded box. From these cycles, the sets generated are given by:  $S_1 = \{6, 7, 8\}$  and  $S_2 = \{3, 4, 7, 8\}$ . The analysis of  $S_1$  identifies arc 8, as the arc for which the removal of its successor does not interfere with other potential *s-t* paths. Thus, arc 6 should not be coded as a successor of arc 8 in the successor matrix. An equivalent analysis for  $S_2$  identifies arc 8, as the arc for which the removal of its successor does not interfere

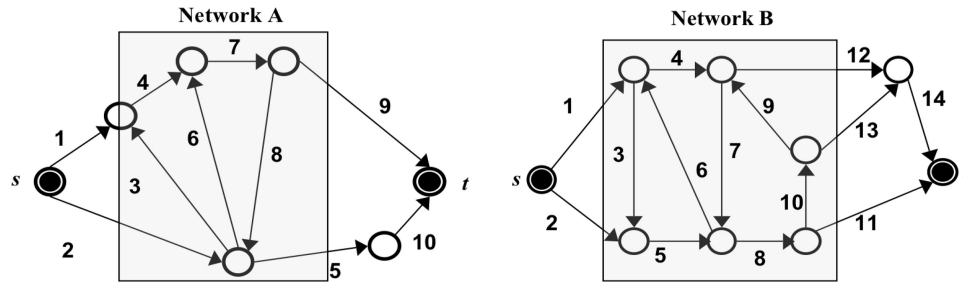


Fig. 4. Cyclic networks A and B.

with other potential  $s-t$  paths. Thus, the successor matrix should not contain arc 3 as a successor of arc 8.

- For network B, the shaded box contains all the arcs that are responsible for generating cycles. The sets generated are:  $S_1 = \{7, 8, 9, 10\}$ ,  $S_2 = \{4, 6, 7\}$  and  $S_3 = \{3, 5, 6\}$ . By analyzing  $S_1$ , arc 9 can be identified as the arc for which the removal of its successor does not interfere with other potential  $s-t$  paths. Thus, arc 7 should not be coded as a successor of arc 9 in the successor matrix. Similarly, the analysis of  $S_2$  identifies arc 7, as the arc for which the removal of its successor does not interfere with other potential  $s-t$  paths. Thus, the successor matrix should not contain arc 6 as a successor of arc 7. Finally,  $S_3$  implies that not including arc 3 as a successor of arc 6 in the corresponding matrix would not interfere with other potential  $s-t$  paths.

## 6.2. Conclusions

This paper examined the  $M2TR_d$  modeling and computation problem. A generalized approach that provides component states that guarantee system success has been proposed. The method has been tested to show that it can effectively solve the problem of generating MMPVs for the  $M2TR_d$  problem. The algorithm uses an information sharing approach similar to that of Ramirez-Marquez *et al.* (2004) that significantly reduces the number of vector analyses needed to obtain all the MMPVs. Moreover, the algorithm mimics natural organisms in the sense that the component levels are inherited from a select number of components in a set called the primary set. The information sharing rationale is complemented by a network reduction technique to obtain the MMPVs.

Although it has been assumed that the nodes are perfectly reliable, the method can be modified to accommodate unreliable nodes. Unlike other approaches, the algorithm can handle both homogeneous and heterogeneous component states. The only restriction on the application of the proposed method is that the network has no cycles. The paper also explains issues related to the reliability computation for the  $M2TR_d$  based on MMPVs.

The MMPV computational approach presented in this paper is preferable when considering sparse large networks.

For dense networks, the MMCV algorithm of Ramirez-Marquez *et al.* (2004) can be used as an alternative M2TR computation method. Finally, by complementing a multistate network MMCV with MMPVs useful information (i.e., criticality analysis, importance and sensitivity measures) regarding the network functioning can be gained.

## References

- Aven, T. (1985) Reliability evaluation of multistate systems with multistate components. *IEEE Transactions on Reliability*, **34**(5), 473–479.
- Aven, T. and Østebø, R. (1986) Two new importance measures for a flow network system. *Reliability Engineering*, **14**, 75–80.
- Ball, M. and Provan, S. (1983) Calculating bounds on reachability and connectedness in stochastic networks. *Networks*, **13**, 253–278.
- Ball, M. and Van Slyke, R. (1977) Backtracking algorithms for network reliability analysis. *Annals of Discrete Mathematics*, **1**, 49–64.
- Barlow, W. and Wu, A. (1978) Coherent systems with multistate components. *Mathematics of Operations Research*, **3**(4), 275–281.
- Billinton, R. and Zhang, W. (2000) State extension for adequacy evaluation of composite power systems-applications. *IEEE Transactions on Power Systems*, **15**(1), 427–432.
- Block, H. and Savits, T. (1982). A decomposition for multistate monotone systems. *Journal of Applied Probability*, **19**, 391–402.
- Boedigheimer, R. and Kapur, K. (1994) Customer-driven reliability models for multistate coherent systems. *IEEE Transactions on Reliability*, **43**(1), 46–50.
- Colbourn, C. (1988) Edge-packings of graphs and network reliability. *Discrete Mathematics*, **72**, 49–61
- Dotson, W. and Gobien, J. (1979) A new analysis technique for probabilistic graphs. *IEEE Transactions on Circuits and Systems*, **26**(10), 855–865.
- El-Newehi, E., Proschan, F. and Sethuraman, J. (1978) Multistate coherent systems. *Journal of Applied Probability*, **15**, 675–688.
- Elsayed, E. (1996) *Reliability Engineering*, Addison Wesley Longman, New York.
- Esary, J. and Proschan, F. (1963) Coherent structures of non-identical components. *Technometrics* **5**(2), 191–209.
- Fishman G., (1986) A comparison of four Monte Carlo methods for estimating the probability of  $s-t$  connectedness. *IEEE Transactions on Reliability*, **35**, 145–155.
- Fotuhi-Firuzabad, M., Billinton, R., Munian, T. and Vinayagam, B. (2004) “A novel approach to determine minimal tie-sets of complex network.” *IEEE Transactions on Reliability*, **53**, 61–70.
- Griffith, W. (1980) Multistate reliability models. *Journal of Applied Probability*, **17**, 735–744.
- Hansler, E. (1972) A fast recursive algorithm to calculate the reliability of a communication network. *IEEE Transactions on Communications*, **20**(3), 637–640.

- Hudson, J. and Kapur, K. (1983) Reliability analysis for multistate systems with multistate components. *IIE Transactions*, **15**(2), 127–134.
- Jin, T. and Coit, D. (2003) Network reliability estimates using linear and quadratic unreliability of minimal cuts. *Reliability Engineering & System Safety*, **48**(3), 234–246.
- Kuo, S., Lu, S. and Yeh, F. (1999) Determining terminal pair reliability based on edge expansion diagrams using OBDD. *IEEE Transactions on Reliability*, **48**(3), 234–246.
- Levitin, G. and Lisnianski, A. (2003) *Multi-State System Reliability*. World Scientific Publishing, Singapore.
- Levitin, G., Lisnianski, A., Ben-Haim, H. and Elmakis, D. (1998) Redundancy optimization for series-parallel multi-state systems. *IEEE Transactions on Reliability*, **47**(2), 165–172.
- Levitin, G., Podofillini, L. and Zio, E. (2003) Generalised importance measures for multi-state elements based on performance level restrictions. *Reliability Engineering & System Safety*, **82**, 287–298.
- Lin J., Jane C. and Yuan J. (1995) On reliability evaluation of a capacitated flow network in terms of minimal pathsets. *Networks*, **25**, 354–361.
- Lin, Y. (2001) A simple algorithm for reliability evaluation of a stochastic-flow network with node failure. *Computers and Operations Research*, **28**, 1277–1285.
- Lin, Y. (2002) Using minimal cuts to evaluate the system reliability of a stochastic-flow network with failures at nodes and arcs. *Reliability Engineering & System Safety*, **75**, 41–46.
- Natvig, B. (1982) Two suggestions on how to define a multistate coherent system. *Advances in Applied Probability*, **14**, 434–455.
- Natvig, B., Sørmo, S., Holen, A. and Høgåsen, G. (1986) Multistate reliability theory—a case study. *Advances in Applied Probability*, **18**, 921–932.
- Netes, V. and Filin, B. (1996) Consideration of node failures in network-reliability calculation. *IEEE Transactions on Reliability*, **45**(1), 127–128.
- Patra, S. and Misra, B. (1993) Reliability evaluation of flow networks considering multistate modeling of network elements. *Microelectronics and Reliability*, **33**(14), 2161–2164.
- Patra, S. and Misra, B. (1996) Evaluation of probability mass function of flow in a communication network considering a multistate model of network links. *Microelectronics and Reliability*, **36**(3), 415–421.
- Ramirez-Marquez, J.E. and Coit, D. (2003) Alternative approach for analyzing multistate network reliability. *Proceedings of the Industrial Engineering Research Conference (IERC)*, Portland, OR, 2003.
- Ramirez-Marquez, J.E. and Coit, D. (2004) A heuristic for solving the redundancy allocation problem for multistate series-parallel systems. *Reliability Engineering & System Safety*, **83**(3), 341–349.
- Ramirez-Marquez, J.E. and Coit, D. (2005a) A Monte-Carlo simulation approach for approximating multi-state two-terminal reliability. *Reliability Engineering & System Safety*, **87**(2), 253–264.
- Ramirez-Marquez, J. and Coit, D. (2005b) Composite importance measures for multistate systems with multistate components. *IEEE Transactions on Reliability*, **54**(3), 517–529.
- Ramirez-Marquez, J.E., Coit, D. and Tortorella, M. (2004) Multi-state two-terminal reliability—a generalized cut set approach. IE Working Paper 03-135, Rutgers University, Piscataway, NJ.
- Ross, S. (1979) Multivalued state component systems. *The Annals of Probability*, **7**(2), 379–383.
- Torrieri, D. (1994) Calculation of node-pair reliability in large networks with unreliable nodes. *IEEE Transactions on Reliability*, **43**(3), 375–379.
- Yeh, W. (2004) A simple MC-based algorithm for evaluating reliability of stochastic-flow network with unreliable nodes. *Reliability Engineering & System Safety*, **83**(1), 47–55.

## Biographies

Jose E. Ramirez-Marquez is an Assistant Professor at Stevens Institute of Technology in the Department of Systems Engineering and Engineering Management. His research interests include system reliability and quality assurance, uncertainty modeling, advanced heuristics for system reliability analysis, applied probability and statistical models, and applied operations research. He obtained his Ph.D. at Rutgers University in Industrial and Systems Engineering. He received his B.S. degree in Actuarial Science from the UNAM in Mexico City in 1998 and M.S. degrees in Industrial Engineering and Statistics from Rutgers University. He is a member of IIE, IFORS and INFORMS.

David W. Coit is an Associate Professor in the Department of Industrial & Systems Engineering at Rutgers University. He received a BS degree in mechanical engineering from Cornell University, an MBA from Rensselaer Polytechnic Institute and MS and PhD in industrial engineering from the University of Pittsburgh. He also has over ten years of experience working for IIT Research Institute (IITRI), Rome NY where he was a reliability analyst, project manager, and engineering group manager. In 1999, he was awarded a CAREER grant from NSF to study reliability optimization. His current research involves reliability prediction and optimization, risk analysis, and multi-criteria optimization considering uncertainty. He is a member of IIE and INFORMS.

Michael Tortorella is a Research Professor in the Department of Industrial and Systems Engineering. He came to Rutgers in September 2001 upon retiring as a Distinguished Member of Technical Staff from Bell Labs—Lucent Technologies where he was responsible for research into fundamental system, network, and service reliability engineering tools as well as for management of reliability in such critical projects as the SL-280 undersea cable system. His research interests include stochastic processes and their applications to reliability, life data analysis, and telephone traffic, as well as design for reliability methods and technologies, process science, service and network reliability and performance, numerical methods in operations research and engineering, and stochastic stability.

*Contributed by the Reliability Engineering Department*