IIE Transactions (2007) **39**, 811–817

ISSN: 0740-817X print / 1545-8830 online
DOI: 10.1080/07408170601013653

An efficient method for reliability evaluation of multistate networks given all minimal path vectors

MING J. ZUO[1,*], ZHIGANG TIAN[1] and HONG-ZHONG HUANG[2]

[1] *Department of Mechanical Engineering, University of Alberta, Edmonton, Alberta, Canada, T6G 2G8*
E-mail: ming.zuo@ualberta.ca
[2] *School of Mechanical, Electronic, and Industrial Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, People's Republic of China*
E-mail: hzhuang@uestc.edu.cn

Received September 2005 and accepted August 2006

The multistate networks under consideration consist of a source node, a sink node, and some independent failure-prone components in between the nodes. The components can work at different levels of capacity. For such a network, we are interested in evaluating the probability that the flow from the source node to the sink node is equal to or greater than a demanded flow of d units. A general method for reliability evaluation of such multistate networks is using minimal path (cut) vectors. A minimal path vector to system state d is called a d-MP. Approaches for generating all d-MPs have been reported. Given that all d-MPs have been found, the issue becomes how to evaluate the probability of the union of the events that the component state vector is greater than or equal to at least one of the d-MPs. There is a need for a more efficient method of determining the probability of this union of events. In this paper, we report an efficient recursive algorithm for this union probability evaluation based on the Sum of Disjoint Products (SDP) principle, and name it the Recursive Sum of Disjoint Products (RSDP) algorithm. The basic idea is that, based on the SDP principle and a specially defined "maximum" operator, "$\oplus$", the probability of a union with L vectors can be calculated via calculating the probabilities of several unions with $L-1$ vectors or less. The correctness of RSDP is illustrated. The efficiency of this algorithm is investigated by comparing it with an existing algorithm that is generally accepted to be efficient. It is found that RSDP is more efficient than the existing algorithm when the number of components of a system is not too small. RSDP provides us with an efficient, systematic and simple approach for evaluating multistate network reliability given all d-MPs.

Keywords: Two-terminal networks, multi-state reliability, minimal path vectors, recursive algorithm

Notation

n = the number of components in the network;
M_i = an integer value representing the maximum state or maximum capacity of component i, $M_i \geq 1$, $i = 1, 2, \ldots, N$;
s = the source node;
t = the sink node;
d = the demand of flow from the source node to the sink node;
x_i = a discrete random variable representing the state or the capacity of component i, x_i may take values $0, 1, 2, \ldots, M_i$, $i = 1, 2, \ldots, N$;
$\mathbf{x}$ = $(x_1, x_2, \ldots, x_N)$ (we call $\mathbf{x}$ the component state vector and it represents the states of all failure-prone components (i.e., all components);
$\phi(\mathbf{x})$ = state of the system;
p_{ij} = $\Pr(x_i = j)$, $i = 1, 2, \ldots, N$, $j = 0, 1, 2, \ldots, M_i$, $\sum_{j=0}^{M_i} p_{ij} = 1$;
P_{ij} = $\Pr(x_i \geq j)$, $i = 1, 2, \ldots, N, j = 0, 1, 2, \ldots, M_i$;
$\mathbf{z}^i$ = the ith minimal path vector of the considered multistate network;
$\mathbf{y}^i$ = a general vector with n elements and with index i;
$\mathbf{Y}^{j,i}$ = a vector generated by the "$\oplus$" operator, $\mathbf{Y}^{j,i} = \mathbf{y}^j \oplus \mathbf{y}^i$;
$\mathtt{PrU}(\bullet)$ = the recursive function of the RSDP algorithm;
$\mathtt{TM}_i$ = the ith term in the SDP calculation;
T_1 = The CPU time used by Aven's algorithm;
T_2 = The CPU time used by RSDP;
λ = the ratio T_1/T_2.

1. Introduction

Many network systems, such as power generation and transmission systems, oil and gas supply systems, and

*Corresponding author

communication systems, consist of components which can work at different levels of capacity. These systems are regarded as multistate networks (Lin *et al*., 1995; Kuo and Zuo, 2003; Lisnianski and Levitin, 2003). Reliability is an important index for evaluating the performance of these systems and for making decisions such as maintenance scheduling. We consider a network which satisfies the following assumptions: (i) all nodes are perfect; and (ii) all links are directed and failure prone. The capacity of a link is an independent discrete random variable which may take non-negative integer values following a certain probability distribution. The term "components", which is usually used in multistate system analysis (Kuo and Zuo, 2003; Lisnianski and Levitin, 2003), will be used to refer to these failure-prone "links".

We limit our discussions to two-terminal reliability analysis. This is a classical network reliability problem with a broad range of practical applications (Kuo *et al*., 1999; Ramirez-Marquez and Coit, 2005). We are interested in the flow from a single source node, s, to a single sink node, t. Under these assumptions, we can call the flow from node s to node t the capacity or the state of the system, represented by $\phi(\mathbf{x})$ where $\mathbf{x}$ is the component state vector. For such a network, we are interested in evaluating the probability that the system state is equal to or greater than d units, i.e., $\phi(\mathbf{x}) \geq d$, which can also be considered to be the reliability of the network once d is specified.

A general method for this multistate network reliability evaluation is using minimal path (cut) vectors. A component state vector, $\mathbf{x}$, is called a minimal path vector to system state d if $\phi(\mathbf{x}) \geq d$, and $\phi(\mathbf{y}) < d$ for any $\mathbf{y} < \mathbf{x}$. Such a minimal path vector is also called a d-MP for short. For the purpose of evaluating the probability of event $\phi(\mathbf{x}) \geq d$, Xue (1985) reports an algorithm for generating all d-MPs. Lin *et al*. (1995) propose another algorithm for the same purpose, claiming it to be more efficient. Using either algorithm, one can find all d-MPs. Let us suppose that there are L such d-MPs and, for simplicity, denote them as $\mathbf{z}^1, \mathbf{z}^2, \ldots, \mathbf{z}^L$. Then, the probability that $\phi(\mathbf{x}) \geq d$ can be calculated is as follows:

$$\Pr(\phi(\mathbf{x}) \geq d) = \Pr(\{\mathbf{x} \geq \mathbf{z}^1\} \cup \{\mathbf{x} \geq \mathbf{z}^2\} \cup \cdots \cup \{\mathbf{x} \geq \mathbf{z}^L\}). \tag{1}$$

Given that all d-MPs have been found, the issue becomes how to evaluate the probability of the union of the events for which the component state vector is greater than or equal to at least one of the d-MPs, as shown in Equation (1). Hudson and Kapur presented methods using the Inclusion-Exclusion (IE) principle and the Sum of Disjoint Products (SDP) principle to evaluate system reliability and reliability bounds for multistate systems given all minimal path vectors or minimal cut vectors (Hudson and Kapur, 1983a, 1983b; Hudson and Kapur, 1985). However, these methods are not systematic and not efficient. Aven (1985) proposed an algorithm based on state space decomposition, which provides a systematic way of evaluating the union probability no matter how many d-MPs exist. It has been proved to be much more efficient than the IE method (Aven, 1985), except for a situation in which the number of d-MPs is much smaller than the number of components, which exists in very few real systems.

In this paper, we propose an efficient recursive algorithm for the evaluation of multistate network reliability based on the SDP principle, and name it Recursive Sum of Disjoint Products (RSDP) algorithm. The basic idea is that, based on the SDP principle and a specially defined "maximum" operator, "$\oplus$", the probability of a union with L vectors can be calculated via calculating the probabilities of several unions with $L-1$ vectors or less. The correctness and efficiency of this algorithm will be investigated.

2. Some definitions

Some definitions that will be used in the proposed RSDP algorithm are presented in this section.

Definition 1. Event $\{\mathbf{x} \geq \mathbf{y}\}$ means $x_i \geq y_i$ for all i, where $\mathbf{x}$ and $\mathbf{y}$ are vectors with the same length.

Consider a multistate network with n independent components. Component i ($1 \leq i \leq n$) has $M_i + 1$ discrete and mutually exclusive states $0, 1, \ldots, M_i$. For a certain vector $\mathbf{y}$ in which y_i represents the state of component i, we have:

$$\Pr(\mathbf{x} \geq \mathbf{y}) = \prod_{i=1}^{n} \Pr(x_i \geq y_i) = \prod_{i=1}^{n} P_{i,y_i}, \tag{2}$$

where P_{i,y_i} is the probability of component i in state y_i or above, $0 \leq y_i \leq M_i$. That is

$$P_{i,y_i} = \sum_{j=y_i}^{M_i} p_{i,j},$$

where M_i represents the highest state of component i, and $p_{i,j}$ is the probability of component i in state j.

Definition 2. Event $\{\mathbf{x} \not\geq \mathbf{y}\}$ is the complement of event $\{\mathbf{x} \geq \mathbf{y}\}$.

Definition 3. A special "maximum" operator, "$\oplus$", is defined as

$$\mathbf{y}^1 \oplus \mathbf{y}^2 \equiv \left(\max\left(y_j^1, y_j^2\right)\right),\ 1 \leq j \leq n. \tag{3}$$

For example, if $\mathbf{y}^1 = (1, 2, 3, 4)$, and $\mathbf{y}^2 = (4, 3, 2, 1)$, we will have $\mathbf{y}^1 \oplus \mathbf{y}^2 = (4, 3, 3, 4)$. The "$\oplus$" operator defined in Equation (3) is designed to manipulate those d-MPs, and it plays an important role in the proposed RSDP algorithm.

3. The RSDP algorithm

3.1. *The proposed RSDP algorithm*

In this section, we will propose a recursive algorithm based on the SDP principle for the evaluation of multistate network reliability. We call this proposed algorithm the RSDP algorithm.

Suppose we have three minimal path vectors $\mathbf{z}^1$, $\mathbf{z}^2$ and $\mathbf{z}^3$. From the SDP principle (Hudson and Kapur, 1983; Kuo and Zuo, 2003), we have:

$$\begin{aligned}&\Pr(\{\mathbf{x}\geq\mathbf{z}^1\}\cup\{\mathbf{x}\geq\mathbf{z}^2\}\cup\{\mathbf{x}\geq\mathbf{z}^3\})\\&\quad=\Pr(\mathbf{x}\geq\mathbf{z}^1)+\Pr(\{\mathbf{x}\not\geq\mathbf{z}^1\}\{\mathbf{x}\geq\mathbf{z}^2\})\\&\quad+\Pr(\{\mathbf{x}\not\geq\mathbf{z}^1\}\{\mathbf{x}\not\geq\mathbf{z}^2\}\{\mathbf{x}\geq\mathbf{z}^3\}).\end{aligned}\tag{4}$$

The first term can be calculated directly using Equation (2). The second term is the probability that events $\{\mathbf{x}\geq\mathbf{z}^2\}$ and $\{\mathbf{x}\not\geq\mathbf{z}^1\}$ occur simultaneously. To evaluate the second term, we can use the following basic probability formula:

$$\Pr(\bar{A}B)=\Pr(B)-\Pr(AB),\tag{5}$$

where A and B are two arbitrary events. Applying Equation (5), the second term in Equation (4) can be written as

$$\begin{aligned}\mathtt{Term2}&=\Pr(\overline{\mathbf{x}\geq\mathbf{z}^1}\{\mathbf{x}\geq\mathbf{z}^2\})\\&=\Pr(\mathbf{x}\geq\mathbf{z}^2)-\Pr(\{\mathbf{x}\geq\mathbf{z}^1\}\{\mathbf{x}\geq\mathbf{z}^2\})\\&=\Pr(\mathbf{x}\geq\mathbf{z}^2)-\Pr(\mathbf{x}\geq(\mathbf{z}^1\oplus\mathbf{z}^2))\\&=\Pr(\mathbf{x}\geq\mathbf{z}^2)-\Pr(\mathbf{x}\geq\mathbf{Y}^{1,2})\end{aligned}\tag{6}$$

where $\mathbf{Y}^{1,2}=\mathbf{z}^1\oplus\mathbf{z}^2$. As shown in Equation (6), the probability of an event involving two vectors, $\mathbf{z}^1$ and $\mathbf{z}^2$, is calculated via the probability of an event involving only one vector, $\mathbf{z}^2$, and the probability of another event involving only one vector, $\mathbf{Y}^{1,2}$.

The third term in Equation (4) is the probability that events $\{\mathbf{x}\geq\mathbf{z}^3\}$, $\{\mathbf{x}\not\geq\mathbf{z}^1\}$ and $\{\mathbf{x}\not\geq\mathbf{z}^2\}$ occur simultaneously. Let $\mathbf{Y}^{1,3}=\mathbf{z}^1\oplus\mathbf{z}^3$, and $\mathbf{Y}^{2,3}=\mathbf{z}^2\oplus\mathbf{z}^3$. Similarly, we can find that the third term "Term3" can be represented as

$$\mathtt{Term3}=\Pr(\mathbf{x}\geq\mathbf{z}^3)-\Pr\Big(\{\mathbf{x}\geq\mathbf{Y}^{1,3}\}\bigcup\{\mathbf{x}\geq\mathbf{Y}^{2,3}\}\Big)\tag{7}$$

As can be seen, the third term can be calculated via the probability of a union of two events. From this specific example, we find that a recursive algorithm based on the SDP principle is viable.

Now we will consider the general case. Suppose we have L general vectors $\mathbf{y}^1,\mathbf{y}^2,\ldots,\mathbf{y}^L$. We define the recursive function as

$$\mathtt{PrU}(\mathbf{y}^1,\mathbf{y}^2,\ldots,\mathbf{y}^L)\equiv\Pr\Big(\bigcup_{i=1}^{L}\{\mathbf{x}\geq\mathbf{y}^i\}\Big)\tag{8}$$

From the SDP principle, we develop the following recursive algorithm:

$$\mathtt{PrU}(\mathbf{y}^1,\mathbf{y}^2,\ldots,\mathbf{y}^L)=\sum_{i=1}^{L}\mathtt{TM}_i,\tag{9}$$

where $\mathtt{TM}_i$ is the ith term in the SDP calculation.

$$\mathtt{TM}_1=\Pr(\mathbf{x}\geq\mathbf{y}^1),\tag{10}$$

$$\begin{aligned}\mathtt{TM}_i&=\Pr(\mathbf{x}\geq\mathbf{y}^i)-\Pr\Big(\bigcup_{j=1}^{i-1}\{\mathbf{x}\geq\mathbf{Y}^{j,i}\}\Big)\\&=\Pr(\mathbf{x}\geq\mathbf{y}^i)-\mathtt{PrU}(\mathbf{Y}^{1,i},\ldots,\mathbf{Y}^{i-1,i}),\quad\mathtt{for}\quad i\geq2,\end{aligned}\tag{11}$$

where vector $\mathbf{Y}^{j,i}=\mathbf{y}^j\oplus\mathbf{y}^i$. The length of $\mathbf{Y}^{j,i}$ is the same as $\mathbf{y}^j$ and $\mathbf{y}^i$, and the value of an element of $\mathbf{Y}^{j,i}$ refers to the corresponding state of the corresponding component. Therefore, from Equations (9), (10) and (11), the $\mathtt{PrU}(\bullet)$ function with L input vectors can be calculated via $\mathtt{PrU}(\bullet)$ functions with $L-1$ input vectors or fewer.

The boundary condition is $L=1$. In this case:

$$\mathtt{PrU}(\bullet)=\mathrm{TM}_1=\Pr(\mathbf{x}\geq\mathbf{y}^1).\tag{12}$$

A simplifying procedure is used in RSDP to reduce, whenever possible, the number of input vectors in function $\mathtt{PrU}(\bullet)$. That is, for any input vector, $\mathbf{y}^i$, if there is an input vector $\mathbf{y}^j$ $(j\neq i)$ satisfying $\mathbf{y}^i\geq\mathbf{y}^j$, $\mathbf{y}^i$ will be deleted from the set of input vectors. The reason for this is that, under the assumption that $\mathbf{y}^i\geq\mathbf{y}^j$, we have $(\mathbf{x}\geq\mathbf{y}^i)\cup(\mathbf{x}\geq\mathbf{y}^j)=(\mathbf{x}\geq\mathbf{y}^j)$.

For a multistate network with L d-MPs $\mathbf{z}^1,\mathbf{z}^2,\ldots,\mathbf{z}^L$, its reliability with respect to level d is

$$\Pr(\phi(\mathbf{x})\geq d)=\mathtt{PrU}(\mathbf{z}^1,\mathbf{z}^2,\ldots,\mathbf{z}^L).$$

3.2. *An illustrative example*

The detailed implementation of RSDP will be illustrated in this section, using the example given by Lin *et al.* (1995). The network system under investigation is a bridge network as shown in Fig. 1. This network has six components (links) represented by C1, C2, ..., C6, respectively. The components are made up from various possible states. The state distributions of the components (Lin *et al.*, 1995) are listed in Table 1.

There are three 3-MPs:

$$\begin{aligned}\mathbf{z}^1&=(3,\ 2,\ 1,\ 0,\ 0,\ 1),\\\mathbf{z}^2&=(2,\ 2,\ 0,\ 0,\ 1,\ 1),\\\mathbf{z}^3&=(2,\ 1,\ 1,\ 0,\ 1,\ 2).\end{aligned}\tag{13}$$

Table 1. State distributions of the components in the example network

State	*0*	*1*	*2*	*3*
Component 1	0.05	0.10	0.25	0.60
Component 2	0.10	0.30	0.60	—
Component 3	0.10	0.90	—	—
Component 4	0.10	0.90	—	—
Component 5	0.10	0.90	—	—
Component 6	0.05	0.25	0.70	—

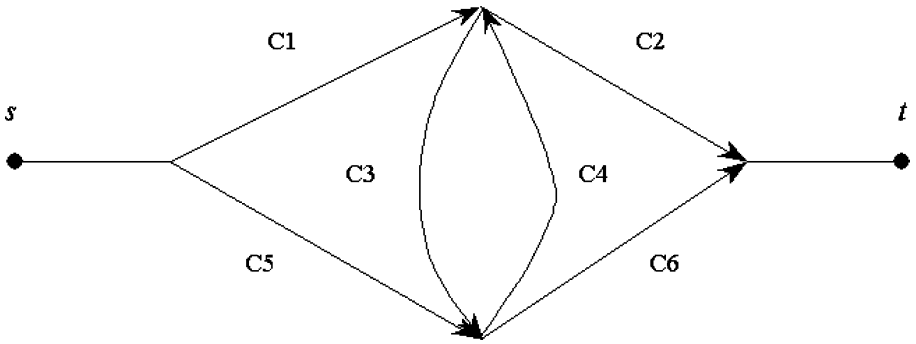

Fig. 1. A multistate network with six components.

The proposed RSDP algorithm is used to calculate $\Pr(\phi(\mathbf{x}) \geq 3)$. From Equation (9) of the RSDP algorithm, there will be three terms in this problem:

$$\begin{aligned}\Pr(\phi(\mathbf{x}) \geq 3) &= \Pr(\{\mathbf{x} \geq \mathbf{z}^1\} \cup \{\mathbf{x} \geq \mathbf{z}^2\} \cup \{\mathbf{x} \geq \mathbf{z}^3\}) \\ &= \mathrm{TM}_1 + \mathrm{TM}_2 + \mathrm{TM}_3. \end{aligned} \tag{14}$$

1. $\mathrm{TM}_1 = \Pr(\mathbf{x} \geq \mathbf{z}^1) = (0.6) \cdot (0.6) \cdot (0.9) \cdot (0.1 + 0.9) \cdot (0.1 + 0.9) \cdot (0.25 + 0.7) = 0.3078.$
2. From Equation (11), we have:

$$\mathrm{TM}_2 = \Pr(\mathbf{x} \geq \mathbf{z}^2) - \Pr(\mathbf{x} \geq \mathbf{Y}^{1,2}), \tag{15}$$

where

$$\mathbf{Y}^{1,2} = \mathbf{z}^1 \oplus \mathbf{z}^2 = (3,\ 2,\ 1,\ 0,\ 1,\ 1).$$

Thus, we get $\mathrm{TM}_2 = 0.1590$.

3. From Equation (11) we obtain:

$$\mathrm{TM}_3 = \Pr(\mathbf{x} \geq \mathbf{z}^3) - \Pr\Big(\{\mathbf{x} \geq \mathbf{Y}^{1,3}\} \bigcup \{\mathbf{x} \geq \mathbf{Y}^{2,3}\}\Big) \tag{16}$$

where

$$\mathbf{Y}^{1,3} = \mathbf{z}^1 \oplus \mathbf{z}^3 = (3,\ 2,\ 1,\ 0,\ 1,\ 2),$$

and

$$\mathbf{Y}^{2,3} = \mathbf{z}^2 \oplus \mathbf{z}^3 = (2,\ 2,\ 1,\ 0,\ 1,\ 2).$$

Since $\mathbf{Y}^{1,3} \geq \mathbf{Y}^{2,3}$, based on "the simplifying procedure", $\mathbf{Y}^{1,3}$ will be removed. Thus, we have:

$$\mathrm{TM}_3 = \Pr(\mathbf{x} \geq \mathbf{y}^3) - \Pr(\mathbf{x} \geq \mathbf{Y}^{2,3}) = 0.1446. \tag{17}$$

Eventually we have:

$$\Pr(\phi(\mathbf{x}) \geq 3) = \mathrm{TM}_1 + \mathrm{TM}_2 + \mathrm{TM}_3 = 0.6114.$$

This result agrees with the result in Lin *et al.* (1995), and this has illustrated the correctness of the proposed RSDP algorithm.

In this example, the RSDP approach evaluates five probability terms. If the IE procedure is used, we need to evaluate seven probability terms. The advantage of the proposed RSDP procedure over the IE principle is clear from the following:

$$\Pr\Big(\{\mathbf{x} \geq \mathbf{Y}^{1,3}\} \bigcup \{\mathbf{x} \geq \mathbf{Y}^{2,3}\}\Big) = \Pr(\mathbf{x} \geq \mathbf{Y}^{2,3}).$$

4. Efficiency investigation of the RSDP

Aven's algorithm (Aven, 1985) is recognized as an efficient reliability evaluation algorithm for multistate systems compared to conventional methods such as the IE method. The computation time with the IE method increases exponentially as the number of MPs, L, increases (Aven, 1985). In this section, we will compare the efficiency of the proposed RSDP algorithm with that of Aven's algorithm. We wrote Aven's algorithm by following the procedure in Aven (1985) and the FORTRAN program in its Appendix. The programs of both RSDP and Aven's algorithm were developed using MATLAB 6.5, and were implemented on a computer with a Pentium M 1.7 GHz CPU and 512 MB of RAM.

In terms of the efficiency of the two algorithms, we are interested in the required computation time with respect to different values for the number of components, n, and different values for the number of MPs, L. First we consider a hypothetical multistate network system with ten components. Each component has ten states, from zero to nine; and the state distributions of all the components are set to be the same. Specifically, the state distribution vector, $\mathbf{p}$, is set to be

$$\mathbf{p} = (0.05,\ 0.15,\ 0.1,\ 0.05,\ 0.15,\ 0.05,\ 0.15,\ 0.1,\ 0.05,\ 0.15).$$

We randomly generate 50 vectors, $\mathbf{z}^1$ to $\mathbf{z}^{50}$, indicating the components' states, and ensure that there are no two vectors $\mathbf{z}^i$ and $\mathbf{z}^j$ satisfying $\mathbf{z}^i \geq \mathbf{z}^j$. That is, no vector is dominated by other vectors in this group (Huang and Zuo, 2004). As a result, these 50 vectors can be treated as all the MPs to a hypothetical system state, say state d. Thus, the probability of the component state vector being not less than any of these hypothetical MPs can be calculated using after RSDP or Aven's algorithm. First, assume that only the first five MPs, $\mathbf{z}^1$ to $\mathbf{z}^5$, are available. RSDP and Aven's algorithm are used respectively to evaluate the probability of interest. Then we change L, the number of MPs, by assuming only

Table 2. Efficiency comparison when the system has ten components

	Number of d-MPs (L)							
	5	*10*	*15*	*20*	*25*	*30*	*40*	*50*
CPU time by Aven's algorithm (T_1)	0.03	0.22	0.44	0.69	1.20	2.15	9.26	15.71
CPU time by RSDP (T_2)	0.01	0.09	0.29	0.83	1.62	2.74	16.475	23.95
Ratio $\lambda = T_1/T_2$	2.60	2.52	1.54	0.83	0.74	0.78	0.56	0.66

Table 3. Efficiency comparison when the system has five components

	Number of d-MPs (L)							
	5	*10*	*15*	*20*	*25*	*30*	*40*	*50*
CPU time by Aven's algorithm (T_1)	0.008	0.023	0.055	0.085	0.13	0.20	0.25	0.29
CPU time by RSDP (T_2)	0.008	0.04	0.10	0.21	0.27	0.35	0.47	0.69
Ratio $\lambda = T_1/T_2$	1.00	0.56	0.55	0.40	0.48	0.56	0.53	0.41

Table 4. Efficiency comparison when the system has 15 components

	Number of d-MPs (L)					
	5	*10*	*15*	*20*	*25*	*30*
CPU time by Aven's algorithm (T_1)	0.11	0.67	2.37	4.97	17.22	35.70
CPU time by RSDP (T_2)	0.01	0.10	0.35	1.39	6.40	17.95
Ratio $\lambda = T_1/T_2$	11.00	6.70	6.76	3.58	2.69	1.99

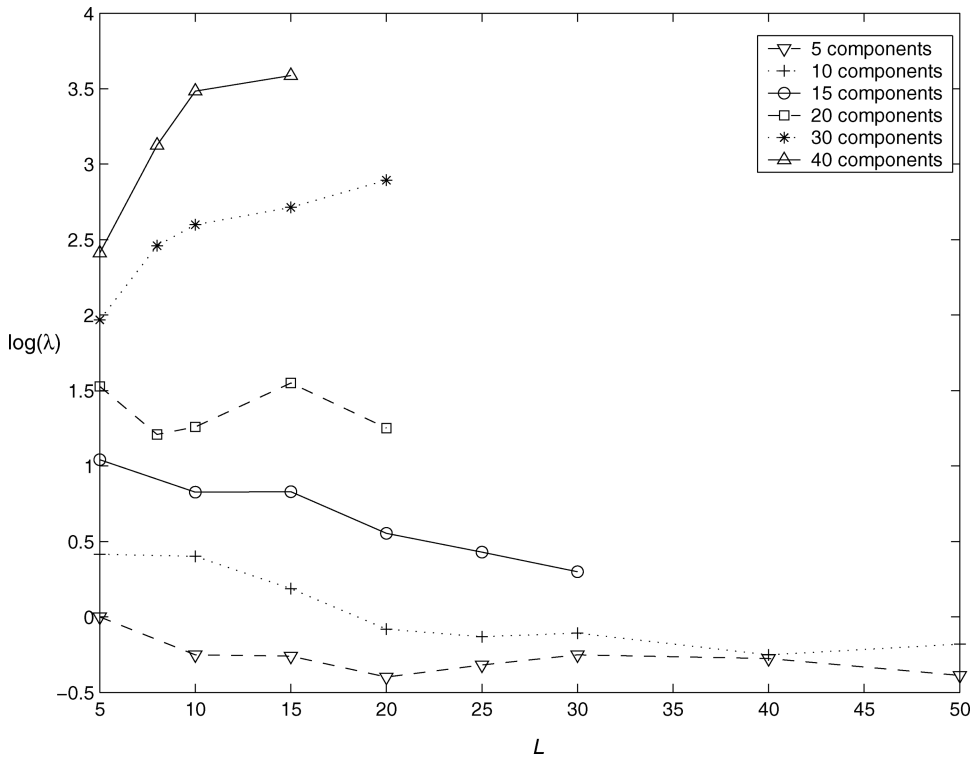

Fig. 2. Ratio λ with respect to different number of components and different *L*.

the first ten, 15, 20, 25 , 30, 40 or 50 MPs are available, respectively, and calculate the same probability of interest. In all these cases, RSDP and Aven's algorithm lead to the same results, which further shows the correctness of RSDP.

We need to note that the 50 MPs used above are generated randomly. The computation time using RSDP or Aven's algorithm may be different if we use another group of 50 MPs. To make more sense in comparing the efficiency of RSDP and Aven's algorithm, we randomly generate ten groups of MP vectors, with 50 MPs in each group. The average CPU time using these ten groups of MP vectors is used to represent the CPU time with respect to a certain L. The CPU time (in seconds) using RSDP or Aven's algorithm is listed in Table 2, where T_1 and T_2 represent the CPU time by Aven's algorithm and by RSDP, respectively. Let $\lambda = T_1/T_2$ denote the ratio between the CPU time of Aven's algorithm and that of RSDP. That is, λ represents the advantage of RSDP over Aven's algorithm in terms of CPU time. If $0 < \lambda < 1$, RSDP is slower than Aven's algorithm; if $\lambda > 1$, RSDP is faster than Aven's algorithm. The bigger the λ is, the more advantageous RSDP is over Aven's algorithm. From Table 2, it appears that the computation time does not increase exponentially as L increases for either RSDP or Aven's algorithm. In this specific network system with ten components, RSDP is a little bit faster when there are 15 MPs or less, while Aven's algorithm is faster when there are 20 MPs or more. From the trend of λ, Aven's algorithm becomes more advantageous with the increase in the number of MP vectors (L) for the network with ten components.

Next, we investigate a multistate network with only five components. The other settings, such as the number of states and the state distribution for each component, remain the same as in the case of a system with ten components. Similarly, we generate ten groups of MP vectors with 50 MPs in each group, and investigate the efficiency of RSDP and Aven's algorithm. The results are shown in Table 3. We find that for the network with five components, Aven's algorithm is faster than RSDP. There is no clear trend for λ with respect to the number of d-MPs. We also conclude that for networks with a small number of components (less than ten in this example), Aven's algorithm is more efficient than RSDP.

Table 6. Efficiency comparison when the system has 30 components

	Number of d-MPs (L)				
	5	*8*	*10*	*15*	*20*
CPU time by Aven's algorithm (T_1)	0.93	17.28	75.49	992	15 646
CPU time by RSDP (T_2)	0.01	0.06	0.19	1.92	20.00
Ratio $\lambda = T_1/T_2$	93.00	288.00	397.29	518.02	782.30

Now, we will investigate multistate networks with more than ten components. First, we consider a multistate network with 15 components. All the other settings remain the same as in the case of system with ten components. The results are shown in Table 4. We have also investigated the case of multistate networks with 20, 30 and 40 components respectively in the same way, and the results are listed in Tables 5, 6 and 7. We did not do the case for Aven's algorithm when there are 40 components and 20 d-MPs, because in this case the expected CPU time using Aven's algorithm is around 40 hours and thus determining the average CPU time would take quite a few days.

In the case of 15 components in Table 4, RSDP is faster than Aven's algorithm with respect to all L values investigated. From the trend of λ, the advantage of RSDP decreases with the increase of L; however, this trend does not exist in the cases of 20, 30 and 40 components, as shown in Tables 5, 6 and 7. In fact, RSDP is much more efficient than Aven's algorithm in these cases. Specifically, RSDP is about 20 times faster than Aven's algorithm in the case of the system with 20 components, and hundreds or thousands of times faster than Aven's algorithm in the cases of the systems with 30 and 40 components. Thus, it seems that the efficiency of Aven's algorithm is much more sensitive to the number of components than that of RSDP. We can conclude that RSDP is more efficient than Aven's algorithm when the number of components of a system is not too small (say, when there are more than 15 components in the system), and the advantage of RSDP becomes stronger with the increase of the number of components in the system.

Table 5. Efficiency comparison when the system has 20 components

	Number of d-MPs (L)				
	5	*8*	*10*	*15*	*20*
CPU time by Aven's algorithm (T_1)	0.31	0.97	3.27	33.55	114.53
CPU time by RSDP (T_2)	0.01	0.06	0.18	0.95	6.42
Ratio $\lambda = T_1/T_2$	31.27	16.17	18.17	35.44	17.84

Table 7. Efficiency comparison when the system has 40 components

	Number of d-MPs (L)				
	5	*8*	*10*	*15*	*20*
CPU time by Aven's algorithm (T_1)	2.58	80.2	670	8698	—
CPU time by RSDP (T_2)	0.01	0.06	0.22	2.25	37.80
Ratio $\lambda = T_1/T_2$	258.00	1336.67	3045.45	3865.96	—

In Fig. 2, we present the ratio λ with respect to different number of components and different number of minimal path vectors L in a graphical manner. The vertical axis represents the ratio λ in a logarithm format. Again, it is very obvious that the advantage of RSDP over Aven's method becomes more significant with the increase of the number of components in the system.

5. Conclusions

In this paper, we developed the RSDP algorithm, an efficient recursive algorithm based on the SDP principle for reliability evaluation of multistate networks. Based on the SDP principle and a specially defined "maximum" operator, "$\oplus$", RSDP can calculate the probability of a union with L vectors via calculating the probabilities of several unions with $L-1$ vectors or fewer. The implementation of RSDP has been illustrated using a simple example. The efficiency of this algorithm has been investigated by comparing it with Aven's algorithm which is widely recognized as being efficient. It has been found that RSDP is more efficient than Aven's algorithm when the number of components of a system is not too small (say, greater than 15), and it can efficiently deal with the reliability evaluation of complex systems with a large number of components. RSDP provides us with an efficient, systematic and simple approach for evaluating multistate network reliability when all d-MPs are given.

Acknowledgements

This research work was supported by the Natural Sciences and Engineering Research Council of Canada, the Alberta Ingenuity Fund, and the Killam Trusts. The constructive comments from all reviewers are very much appreciated.

References

Aven, T. (1985) Reliability evaluation of multistate systems with multistate components. *IEEE Transactions on Reliability*, **34**, 473–479.

Huang, J. and Zuo, M.J. (2004) Dominant multi-state systems. *IEEE Transactions on Reliability*, **53**, 362–368.

Hudson, J.C. and Kapur, K.C. (1983a) Reliability analysis for multistate systems with multistate components. *IIE Transactions*, **15**, 127–135.

Hudson, J.C. and Kapur, K.C. (1983b) Modules in coherent multistate systems. *IEEE Transactions on Reliability*, **32**, 183–185.

Hudson, J.C. and Kapur, K.C. (1985) Reliability bounds for multistate systems with multistate components. *Operations Research*, **33**, 153–160.

Kuo, S., Lu, S. and Yeh, F. (1999) Determining terminal pair reliability based on edge expansion diagrams using OBDD. *IEEE Transactions on Reliability*, **48**, 234–246.

Kuo, W. and Zuo, M.J. (2003) *Optimal Reliability Modeling: Principles and Applications*, Wiley, New York, NY.

Lin, J.S., Jan, C.C. and Yuan, J. (1995) On reliability evaluation of a capacitated-flow network in terms of minimal pathsets. *Networks*, **25**, 131–138.

Lisnianski, A. and Levitin, G. (2003) *Multi-state System Reliability: Assessment, Optimization and Applications*, World Scientific, Singapore.

Ramirez-Marquez, J.E. and Coit, D. (2005) A Monte-Carlo simulation approach for approximating multi-state two-terminal reliability. *Reliability Engineering & System Safety*, **87**, 253–264.

Xue, J. (1985) On multistate system analysis. *IEEE Transactions on Reliability*, **34**, 329–337.

Biographies

Ming J. Zuo is a Professor in the Department of Mechanical Engineering, University of Alberta, Canada. He received a Master of Science degree in 1986 and a Ph.D. degree in 1989 both in Industrial Engineering from Iowa State University, Ames, IA, USA. His research interests include system reliability analysis, maintenance optimization, signal processing, and fault diagnosis. He is a Department Editor of *IIE Transactions*, and an Associate Editor of *IEEE Transactions on Reliability*. He is a senior member of IEEE and IIE.

Zhigang Tian is a Ph.D. student at the Department of Mechanical Engineering, University of Alberta, Canada. He received a B.S. degree in 2000 and an M.S. degree in 2003 both in Mechanical Engineering from Dalian University of Technology, Dalian, China. His research interests include reliability modeling, optimization, and prediction, condition-based maintenance, multi-state reliability, and operations research.

Hong-Zhong Huang is a Professor in the School of Mechatronics Engineering at the University of Electronic Science and Technology of China. He is currently a Visiting Professor at the Department of Mechanical Engineering, University of Alberta, Canada. He received a Ph.D. degree in Reliability Engineering from Shanghai Jiaotong University, Shanghai, China. He has published 120 journal articles and five books in the fields of reliability engineering, optimization design, and fuzzy sets theory. He is a senior member of several professional societies, and has served on the boards of professional societies.