Journal of the Operations Research Society of Japan
Vol. 59, No. 3, July 2016, pp. 258–268

A NEW METHOD OF APPROXIMATION FOR COMPUTING SYSTEM-FAILURE FREQUENCY

Masahiro Hayashi Hisao Yamamoto
Tokyo City University

(Received December 24, 2014; Revised April 22, 2016)

Abstract A method for approximately computing the frequency of system failures is proposed. The method uses multilinear polynomials to compute lower and upper bounds of system availability and transforms these bounds into lower and upper bounds of the system failure frequency without increasing the computational complexity. Most of the currently used high-speed approximation methods that compute the lower and upper bounds of availability use multilinear polynomials, so it is a relatively simple matter to transform these methods into new ones for computing the failure frequency while preserving their speed and accuracy. Experimental results demonstrate that the proposed method can quickly and accurately compute the system failure frequency.

Keywords: Reliability, availability, failure frequency, Birnbaum importance

1. Introduction

Availability is a commonly used reliability measure in the design of telecommunications networks, railways, rockets, etc. Another reliability measure is the failure frequency, but this has received almost no attention except from theoretical researchers.

We have previously shown [1] that failure frequency is actually more valid than availability in practical situations because it has a more direct effect on customer satisfaction. We analyzed the relationship between the occurrences of failures and the market share of several telecommunications service companies and found that the market share decreases of several stem not from problems with availability but rather from the effect of the failure frequency. Specifically, as the failure frequency becomes worse, the market share decreases, while if the availability becomes worse, the market share typically does not decrease. After the results of that study became known, system reliability design using failure frequency became more of a hot topic in the reliability engineering field [2]. The standard design process today typically follows five steps.

Step1 Enumerate possible alternative plans for building a reliable system while considering costs.
Step2 Compute the failure frequency of the system for each of these alternative plans.
Step3 Select the most reliable plan.
Step4 Implement the selected plan.
Step5 Repeat Steps 1–4.

The key to these steps is how to compute the system failure frequency. Theoretical researchers have already proposed various methods to compute the failure frequency [3, 4, 5, 6], but all of them cause exponential increases in computation time when the system size increases. This creates serious difficulties when it comes to computing the failure frequency

for a large system. A practical approach to overcoming these difficulties is constructing high-speed methods of approximation. However, although there are many such methods for availability [3, 4, 5, 6, 7], up to now, there have been none for computing the failure frequency.

In this paper, we propose a new approach to approximating the failure frequency of a system by transforming availability expressed by a multilinear polynomial into failure frequency. By using this transformation, we transform an existing method such as in [7] in order to quickly compute the lower and upper bounds of the failure frequency of a system.

2. Preliminaries

In this paper, we use the following phrases, which are common in the system reliability engineering field:

Up time:	Time from the 'beginning of the system going up' to the 'end of it being up'
Down time:	Time from the 'beginning of the system going down' to the 'end of it being down'

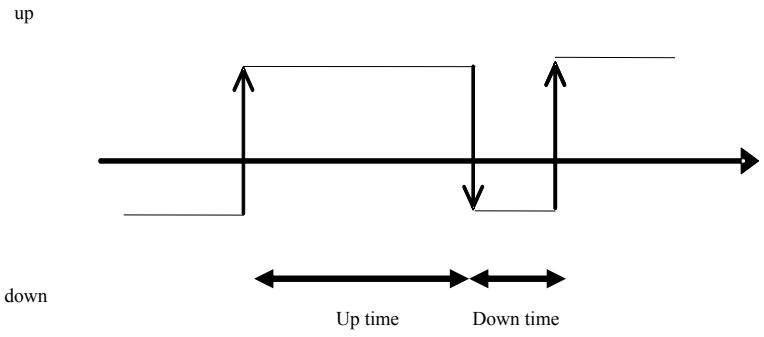

Figure 1: Up time and down time

MTBF:	Average up time
MTTR:	Average down time
Failure rate:	1/MTBF
Repair rate:	1/MTTR
Availability:	MTBF/(MTBF+MTTR)
Unavailability:	MTTR/(MTBF+MTTR)
Failure frequency:	1/(MTBF+MTTR)

The model and reliability measures are described below.

1. Our model is a two-state monotone system [14]. Its structure function is denoted by Ψ where Ψ is a mapping from $\{0,1\}^n$ to $\{0,1\}$. 0 indicates that the system or one of its components has gone down (has failed) and 1 indicates that it is up (working). Let x_i be a variable satisfying $x_i = 1$ if component i is up, and $x_i = 0$ otherwise. The set of states of the components is expressed by a vector $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\} \in \{1,0\}^n$. For $\boldsymbol{x}$, the state of the system is expressed by a function $\Psi(x_1, x_2, \ldots, x_n)$. We call $\boldsymbol{x}$ the state vector.

2. Each component has its own repair personnel. They begin its repair just after a failure occurs and replace it with one as good as new. The lifetime distribution of each component is an exponential distribution with parameter (failure rate) λ_i. The repair time distribution of each component is an exponential distribution with parameter (repair rate) μ_i. That is, the behavior of the state of each component can be described by a two-state continuous-time Markov chain, where these two states are up (1) and down (0). If we describe this Markov chain as $\{X_i(t), t \geq 0\}$, then the stochastic process describing the behavior of the system is $\{\Psi(X_1(t), X_2(t), \ldots, X_n(t)), t \geq 0\}$, which is also a Markov chain.
3. The stationary availability (sometimes we briefly say 'availability') of component i, denoted by A_i, is the probability of component i being up when the system is in a steady state. A_i is computed as

$$A_i = \lim_{t\to\infty} Pr(X_i(t) = 1) = \frac{\frac{1}{\lambda_i}}{\frac{1}{\lambda_i} + \frac{1}{\mu_i}}$$

4. Therefore, the stationary availability of the system denoted by A, can be described using a function h_Ψ, where we call h_Ψ the 'reliability function':

$$A = \lim_{t\to\infty} Pr(\Psi(X_1(t), X_2(t), \ldots, X_n(t)) = 1) = h_\Psi(A_1, \ldots, A_n)$$

The transition rate of the system to down in a steady state is expressed using h_Ψ, as follows.

$$\text{The transition rate of the system in the steady state} = \sum_{i=1}^{n} \left\{ \left(\frac{\partial h_\Psi(A_1, \ldots, A_n)}{\partial A_i} \right) A_i \lambda_i \right\} \quad (1)$$

5. Equation (1) is derived as follows.
The following equations are true because of the well-known total-probability theorem.

$$\begin{aligned} \frac{\partial h_\Psi(A_1, \ldots, A_n)}{\partial A_i} &= \frac{\partial}{\partial A_i}\{h_\Psi(1_i, A_1, \ldots, A_n)A_i + h_\Psi(0_i, A_1, \ldots, A_n)(1 - A_i)\} \quad (2) \\ &= h_\Psi(1_i, A_1, \ldots, A_n) - h_\Psi(0_i, A_1, \ldots, A_n) \quad (3) \end{aligned}$$

In the above, $h_\Psi(1_i, A_1, \ldots, A_n) - h_\Psi(0_i, A_1, \ldots, A_n)$ gives the probability of the system being in a critical condition wherein a single failure of component i will cause it to go down. We can see there are n such critical conditions for $i = 1, 2, \ldots, n$, and these critical conditions are disjoint. Therefore, the transition rate of a system in the steady state from up to down is obtained by summing the transition rates of these critical conditions, i.e., computing Equation (1).
6. This transition rate in the steady-state of the system to down implies the failure frequency of the system [3, 4, 5, 6]. Equation (2) is known as the Birnbaum importance of component i [5].

Failure frequency refers how many failures occur per unit time in the steady state. Even if the failure frequency is small, the availability might still be low when there is a long mean down time for the system.

3. Existing Work

Theoretical researchers have investigated ways of computing the failure frequency of a system. The most relevant work as it relates to our research is summarized in this section.

3.1. Multilinear polynomial for expressing availability

The reliability function $h_\Psi(A_1, \ldots, A_n)$ can be expressed using multilinear polynomials for $A_1, \ldots, A_n$[3], as follows.

Definition 3.1. *Suppose that a polynomial P_o consisting of variables $A_1, \ldots, A_n$ can be expressed as $P_o = a(i)A_i + b(i)$ for any A_i, where $a(i)$ and $b(i)$ are polynomials independent of A_i, then P_o is a multilinear polynomial of $A_1, \ldots, A_n$.*

For simplicity, we will occasionally refer to a multilinear polynomial of $A_1, \ldots, A_n$ as simply a "multilinear polynomial'. The reason $h_\Psi(A_1, \ldots, A_n)$ can be expressed in terms of multilinear polynomials is explained below. As we already said in deriving Equation (2), the total-probability theorem leads to

$$\begin{aligned} h_\Psi(A_1, \ldots, A_n) &= A_i h_\Psi(1_i, A_1, \ldots, A_n) + (1 - A_i) h_\Psi(0_i, A_1, \ldots, A_n) \\ &= \{h_\Psi(1_i, A_1, \ldots, A_n) - h_\Psi(0_i, A_1, \ldots, A_n)\} A_i + h_\Psi(0_i, A_1, \ldots, A_n). \end{aligned}$$

Letting $a(i)$ and $b(i)$ be $h_\Psi(1_i, A_1, \ldots, A_n) - h_\Psi(0_i, A_1, \ldots, A_n)$ and $h_\Psi(0_i, A_1, \ldots, A_n)$, respectively, we have $h_\Psi(A_1, \ldots, A_n) = a(i)A_i + b(i)$, wherein we may notice that $a(i)$ and $b(i)$ do not include A_i, and so are independent of A_i. This is true for any A_i for $i = 1, 2, \ldots, n$. Therefore, $h_\Psi(A_1, \ldots, A_n)$ is a multilinear polynomial.

As an example, let us consider a parallel system composed of two components 1 and 2. We can express the system availability A as $A = A_1 + A_2 - A_1A_2 = (1 - A_2)A_1 + A_2$. Here, if we set $i = 1, a(i) = 1 - A_2$ and $b(i) = A_2$, we can find A of the form $a(i)A_i + b(i)$. Furthermore, $A = (1 - A_2)A_1 + A_2$ is also true for parallel systems, meaning we can also find A of the form $a(i)A_i + b(i)$ by setting $i = 2, a(i) = 1 - A_1$ and $b(i) = A_1$. A parallel system can thus be expressed as a multilinear polynomial of A_1 and A_2.

3.2. Previous work on evaluating the failure frequency

Equation (1) leads to

$$\text{Failure frequency of } S = \sum_{i=1}^{n} \left(\frac{\partial h_\Psi(A_1, \ldots, A_n)}{\partial A_i} \right) A_i \lambda_i \tag{4}$$

Reference [4] defines $D(h_\Psi(A_1, \ldots, A_n))$ as

$$D(h_\Psi(A_1, \ldots, A_n)) = \sum_{i=1}^{n} \left(\frac{\partial h_\Psi(A_1, \ldots, A_n)}{\partial A_i} \right) A_i \lambda_i$$

That is, D simply denotes the failure rate of Equation (1).

Reference [4, 6] show that if we have an algorithm to determine the value of $h_\Psi(A_1, \ldots, A_n)$ from the values of $A_1, \ldots, A_n$ by using addition, subtraction, multiplication, and division, then with minor revision, this algorithm can also be used to determine the value of $D(h_\Psi(A_1, \ldots, A_n))$ from $A_1, \ldots, A_n$ and $\lambda_1, \ldots, \lambda_n$, without any increase in the order of computing complexity.

4. Approximation

The results presented in [4, 6] suggest that a high-speed method for computing the failure frequency $D(h_\Psi(A_1, \ldots, A_n))$ can be constructed if a high-speed method for computing the availability A can be constructed. Many researchers have studied ways of computing the system availability [3, 7, 8, 9, 10] but this problem is known to be NP-hard [3, 13], which means that any existing method for exactly computing the system availability causes an

exponential increase in the computing time if the system is large. The reason this problem is NP-hard is that if we could solve this problem in polynomial time, then we could also solve the problem of counting the number of cutsets in polynomial time, but this counting problem is known to be NP-hard [13].

Therefore, even if we apply the methods of [4, 6] to an existing method of exactly computing A, the time needed for computing $D(h_\Psi(A_1, \ldots, A_n))$ would exponentially increase. A practical approach to overcoming this difficulty is to construct high-speed methods of approximation, and such practical methods have already been proposed for computing A [3, 7]. These methods can compute the lower and upper bounds for A, thereby enabling us to estimate the errors caused by the approximations. Regrettably, when we apply the methods of [4, 6] to the lower and upper bounds of A, we do not always obtain the lower and upper bounds of $D(h_\Psi(A_1, \ldots, A_n))$. For example, if $A_1 = A_2 = 0.999$, then $A_1A_2 \leq A_1+A_2-A_1A_2$ because $A_1A_2 = 0.998001$ and $A_1 + A_2 - A_1A_2 = 0.999999$. However, if $A_1 = A_2 = 0.999$ and $\lambda_1 = \lambda_2 = 0.2$ then $D(A_1A_2) \geq D(A_1 + A_2 - A_1A_2)$ because $D(A_1A_2) = 0.39992$ and $D(A_1+A_2-A_1A_2) = 0.0003996$. In this example, A_1A_2 gives the lower bound of availability for a parallel system, but this is not the lower bound of the failure frequency for it. In such case, we cannot guarantee the accuracy of computing $D(h_\Psi(A_1, \ldots, A_n))$. In this sense, therefore, it is difficult to apply an approximation algorithm for computing availability to failure frequency.

Here, we propose a way of overcoming this difficulty, i.e., by deriving an upper bound and lower bound of Equation (1) by using the fact that h_Ψ is a multilinear polynomial. Theorem 4.1 gives the general framework of this new method. The following definitions of $F_{\rm lower}(f_1, f_2)$ and $F_{\rm upper}(f_1, f_2)$ are necessary for describing Theorem 4.1.

$$F_{\rm lower}(f_1, f_2) = \mathrm{Max}(D(f_1) - (f_2 - f_1)\sum_{i=1}^{n}\mu_i, D(f_2) - (f_2 - f_1)\sum_{i=1}^{n}\lambda_i) \quad (5)$$

$$F_{\rm upper}(f_1, f_2) = \mathrm{Max}(D(f_1) + (f_2 - f_1)\sum_{i=1}^{n}\lambda_i, D(f_2) + (f_2 - f_1)\sum_{i=1}^{n}\mu_i) \quad (6)$$

See Section 2 for the failure rate λ_i and repair rate μ_i in these definitions.

Theorem 4.1. *If f_1, f, and f_2 satisfy the following conditions*:
(i) *f_1, f, and f_2 are multilinear polynomials for $A_1, \ldots, A_n$;*
(ii) *$f_1 \leq f \leq f_2$ when $0 \leq A_1 \leq 1, 0 \leq A_2 \leq 1, \ldots,$ and $0 \leq A_n \leq 1$;*
then $F_{lower}(f_1, f_2) \leq D(f) \leq F_{upper}(f_1, f_2)$.

The details of the proof are in Appendix 1. From this theorem, when f_1 gives the lower bound of the availability of the system and when f_2 gives the upper bound of S, $F_{\rm lower}(f_1, f_2)$ gives the lower bound of the failure frequency of the system and $F_{\rm upper}(f_1, f_2)$ gives the upper bound of the failure frequency of S. It is easy to see that the right sides of equations (5) and (6) consist of negative and positive numbers, summations up to n, and computations of f_1 and f_2. Accordingly, we can easily derive the following theorem.

Theorem 4.2. *The order of complexity of computing f_1 and f_2 is the same as the order of complexity of computing $F_{lower}(f_1, f_2)$* (or $F_{\rm upper}(f_1, f_2)$).

The details of the proof are also in Appendix 1. By applying these theorems to the case that $f = h_\Psi(A_1, \ldots, A_n)$, if we have a high-speed method of approximation using multilinear polynomials to determine the lower and upper bounds for system availability $h_\Psi(A_1, \ldots, A_n)$, this method can be transformed into one for computing the lower and

upper bounds for the failure frequency $D(h_\Psi(A_1, \ldots, A_n))$ without increasing the order of computational complexity.

Remark 4.1. We can derive the following inequation.

$$F_{upper}(f_1, f_2) - F_{lower}(f_1, f_2) \leq (f_2 - f_1) \left(\sum_{i=1}^{n} \lambda_i + \sum_{i=1}^{n} \mu_i \right) \tag{7}$$

The proof of this inequation is shown in Appendix 2. This inequation guarantees that if f_1 and f_2 become close in value, the failure frequencies of $F_{\text{lower}}(f_1, f_2)$ and $F_{\text{upper}}(f_1, f_2)$ also become close.

Remark 4.2. Theorem 4.1 is valid when we have the lower bound f_1 and upper bound f_2 of f, where f_1 and f_2 are multilinear polynomials. How to obtain these bounds is important. Carlier [7] gives a method to obtain them, and it is briefly explained in Appendix 3.

5. Numerical Examples

We wrote software to compute the lower and upper bounds of the system failure frequency of telecommunications networks by using the equations in Theorem 4.1. Two examples of such computations are presented in this section. Figures 3 and 4 are overviews of the systems, where each path is represented by an arrow. These examples have similar topologies but different numbers of nodes, links, and paths. Each path works if and only if all nodes and links through which the path goes are up. The minimum number of working paths for the system to be up is 12 in Figure 2, and 17 in Figure 3. Each failure is repaired by its own repairer.

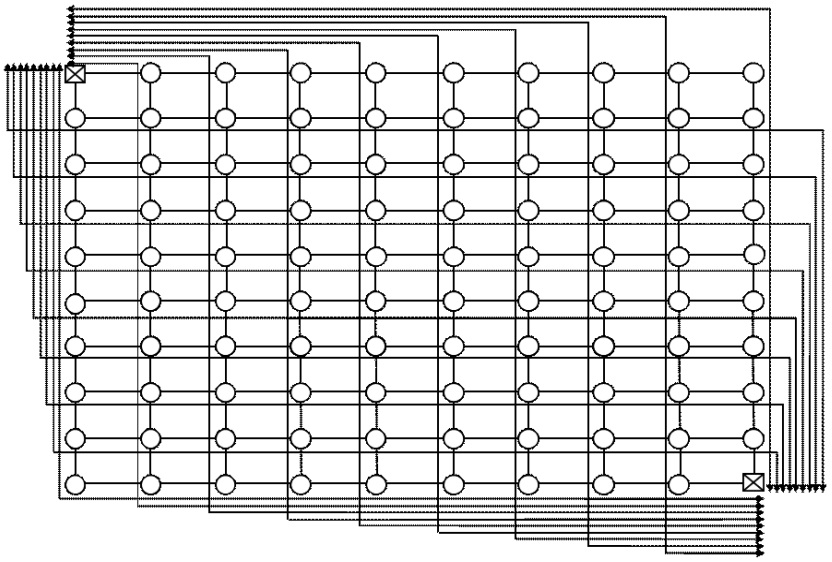

Figure 2: Example 5.1

The parameters used in the evaluation are presented below.

Example 5.1. The availability of each node was 0.99999, and the failure rate of each node was 1.0×10^{-6} (per hour). The availability of each vertical link was 0.9995, and the failure rate of each vertical link was 2.0×10^{-7}(per hour). The availability of each horizontal link was 0.9999, and the failure rate of each horizontal link was 1.0×10^{-7} (per hour).

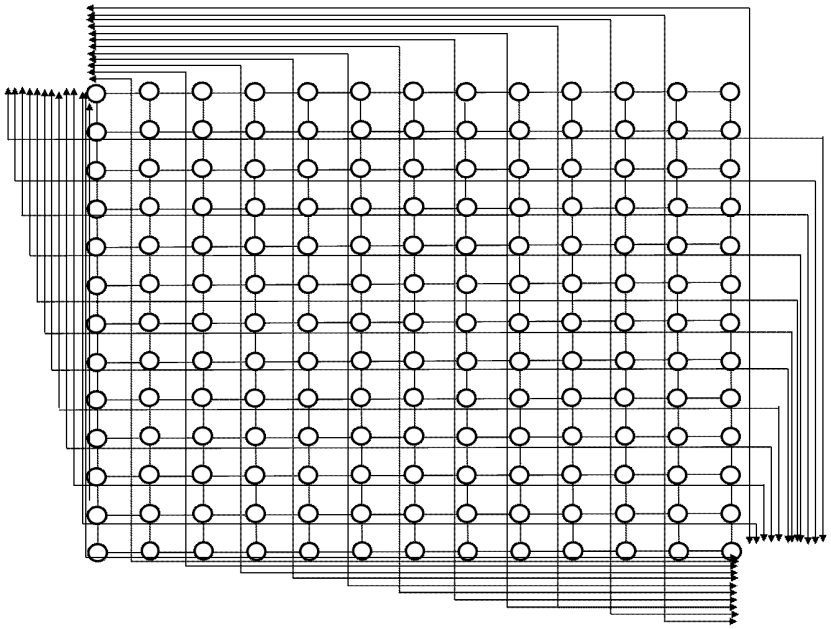

Figure 3: Example 5.2

Example 5.2. The availability of each node was 0.9999, and the failure rate of each node was 1.0×10^{-1} (per hour). The availability of each link was also 0.9999, and the failure rate of each link was 1.0×10^{-1} (per hour).

Tables 1 and 2 summarize the numerical results.

Table 1: Numerical results of approximating the failure frequency of Example 5.1.

	Lower bound Upper bound	Exact value
Computed results(/hour)	2.61964×10^{-4} 2.61989×10^{-4}	2.61970×10^{-4}
Computing time (s)	14.6	523

Table 2: Numerical results of approximating the failure frequency of Example 5.2.

	Lower bound Upper bound	Exact value
Computed results(/hour)	4.25452×10^{-1} 4.25483×10^{-1}	We tried to compute it
Computing time (s)	60.3	in 10 days but did not succeed.

The exact values in Table 1 were computed by applying [6, 11]. While we also tried to compute the exact value for Example 5.2 by applying [6, 11], the computation still hadn't finished after ten days. We estimate that the computation time for Example 5.2 would be astronomical. We used Python 3.1.1 as the programming language, Windows XP as the OS, and an Intel Pentium 3.4 GHz with a 1-GB memory as the CPU. These results clearly demonstrate that the proposed transformed method is fast and accurate. The algorithm presented in [11] is based on the path-set approach. Other approaches, such as the cutset approach, were not tested with these numerical examples. However, the exact evaluation problem is known to be NP-hard. Therefore, the computation times for an exact evaluation would be very long for these models.

6. Conclusion

We proposed an approximation method for computing the failure frequency of a system. Our basic premise was that if we start with a high-speed approximation using multilinear polynomials to derive lower and upper bounds for the system availability, we can then transform the method into one for computing lower and upper bounds for the system failure frequency without increasing the computational complexity. The results of numerical experiments on this method demonstrated that it could deliver very high speed and great accuracy.

We intend to undertake three projects as future work.

1. Improve the method to make it even faster and more accurate.
2. Apply the method to other key measures such as the frequency of time-specific failures [4].
3. Apply the method to the design of actual large systems.

References

[1] M. Hayashi: Effects of reliability measures on market share. *The IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, **E94–A 10** (2011), 2043–2047.

[2] M. Hayashi: A new topic in reliability of telecommunications networks. *Reliability Engineering Association of Japan*, **33 5** (2011), 230–235.

[3] W.G. Shneeweiss: *Boolean Functions with Engineering Applications and Computer Programs* (Springer-Verlag, Berlin, 1989).

[4] M. Hayashi: System failure-frequency analysis using a differential operator. *IEEE Transaction on Reliability*, **R–40 5** (1991), 601-614.

[5] Y.R. Chang, S.V. Amari, and S.Y. Kuo: Computing system failure frequency and reliability importance measures using OBDD. *IEEE Transaction on Computers*, **C–53 1** (2004), 54-68.

[6] M. Hayashi and T. Abe, and I. Nakajima: Transformation from availability expression to failure frequency expression. *IEEE Transaction on Reliability*, **R–55 2** (2006), 252-261.

[7] J. Carlier, Y. Li, and J. Lutton: Reliability evaluation of large telecommunication networks. *Discrete Applied Mathematics*, **76** (1997), 61-80.

[8] S.Y. Kuo, F. M. Yeh, and H.Y. Lin: Efficient and exact reliability evaluation for networks with imperfect vertices. *IEEE Transaction on Reliability*, **R–56 2** (2007), 288-300.

[9] A. Satyanarayana and R.K. Wood: A linear time algorithm for computing K-terminal reliability in series and parallel networks. *Society for Industrial and Applied Mathematics Journal on Computing*, **14 4** (1985), 818-832.

[10] G. Chaudohuri, K. Hu, and N. Afshar: A new approach to system reliability. *IEEE Transaction on Reliability*, **R–50 1** (2000), 75-84.

[11] M. Hayashi and T. Abe: Evaluating reliability of telecommunications networks using traffic path information. *IEEE Transaction on Reliability*, **R–57 2** (2008), 283-294.

[12] Y.R. Chang, S.V. Amari, and S.Y. Kuo, Computing system failure frequencies and reliability importance measures using OBDD. *IEEE Transaction on Reliability*, **R–53 1** (2004), 54-68.

[13] J.S. Provan and M.O. Ball: The complexity of counting cuts and computing the probability that a graph is connected. *Society for Industrial and Applied Mathematics Journal on Computing*, **12** (1983), 77-788.

[14] R.E. Barlow and F. Prochanr: *Mathematical Theory of Reliability (Classics in Applied Mathematics)* (Society for Industrial and Applied Mathematics, New York, 1965).

Appendix 1:

Proof of Theorem 4.1 *and* 4.2. To prove Theorem 4.1, we need the following Lemma.

LemmaA1. *For g_1 and g_2 which are multilinear polynomials of $A_1, \ldots, A_n$, if the following is true,*

$$for\ any\ A_1 \in [0,1]\ (i = 1, 2, \ldots, n), g_1(A_1, \ldots, A_n) \leq g_2(A_1, \ldots, A_n) \tag{A.1}$$

then the following is true.

$$-(g_2 - g_1)\sum_{i=1}^{n}\mu_i \leq D(g_2) - D(g_1) \leq (g_2 - g_1)\sum_{i=1}^{n}\lambda_i$$

Proof. g_1 and g_2 are multilinear polynomials. Therefore, we have two polynomials $a_i(i)$, $b_j(i)$, $j = 1, 2$, which are independent of A_i, and these satisfy

$$g_j = a_j(i)A_i + b_j(i), \quad j = 1, 2$$

Therefore,

$$b_2(i) - b_1(i) = (g_2 - g_1) - \left(\frac{\partial g_2}{\partial A_i} - \frac{\partial g_1}{\partial A_i}\right)A_i. \tag{A.2}$$

By (A.1) in Lemma A1, $g_1(0_i, A_1, \ldots, A_n) \leq g_2(0_i, A_1, \ldots, A_n)$. Therefore, $b_1(i) \leq b_2(i)$. Moreover, from (A.2) and the fact that $\lambda_i \geq 0$, we have

$$\left(\frac{\partial g_2}{\partial A_i} - \frac{\partial g_1}{\partial A_i}\right)A_1\lambda_i \leq (g_2 - g_1)\lambda_i. \tag{A.3}$$

Again, by (A.1) in Lemma A1, $g_1(1_i, A_1, \ldots, A_n) \leq g_2(1_i, A_1, \ldots, A_n)$. Therefore, the same logic leads to

$$-(g_2 - g_1)\mu_i \leq \left(\frac{\partial g_2}{\partial A_i} - \frac{\partial g_1}{\partial A_i}\right)(1 - A_i)\mu_i.$$

From the fact that A_i is availability and it satisfies $A_i\lambda_i = (1 - A_i)\mu_i$ (See Section 2), we obtain

$$-(g_2 - g_1)\mu_i \leq \left(\frac{\partial g_2}{\partial A_i} - \frac{\partial g_1}{\partial A_i}\right)A_i\lambda_i. \tag{A.4}$$

(A.3) and (A.4) result in that Lemma A1 is true. □

Now, we can prove Theorem 4.1 using Lemma A1.

Proof of Theorem 4.1. We assumed $f_1 \leq f \leq f_2$ in Theorem 4.1. Therefore, Lemma A1 leads to

$$-(f - f_1)\sum_{i=1}^{n}\mu_i \leq D(f) - D(f_1) \leq (f - f_1)\sum_{i=1}^{n}\lambda_i \tag{A.5}$$

$$(f_2 - f)\sum_{i=1}^{n}\mu_i \le D(f_2) - D(f) \le (f_2 - f)\sum_{i=1}^{n}\lambda_i \tag{A.6}$$

Again, we have used the assumption $f_1 \le f \le f_2, f - f_1 \le f_2 - f_1, f_2 - f \le f_2 - f_1$. Therefore, (A.5) and (A.6) lead to

$$D(f_1) - (f_2 - f_1)\sum_{i=1}^{n}\mu_i \le D(f) \le D(f_1) + (f_2 - f_1)\sum_{i=1}^{n}\lambda_i,$$

$$D(f_2) - (f_2 - f_1)\sum_{i=1}^{n}\lambda_i \le D(f) \le D(f_2) + (f_2 - f_1)\sum_{i=1}^{n}\mu_i.$$

Hence Theorem 1 is proved. □

Proof of Theorem 4.2. As preparation, we define $O()$ as the order of complexity for computing the value in parentheses ( ). Computing $F_{\text{lower}}(f_1, f_2)$ consists of computing $f_1, f_2, D(f_1), D(f_2)$, the sum of λ_i, and the sum of μ_i. Therefore, $O(F_{\text{lower}}(f_1, f_2)) = O(f_1) + O(f_2) + O(D(f_1)) + O(D(f_2))$. Ref. [5] already showed that the computation complexities for f and $D(f)$ are the same for any multlinear polynomial f. Therefore, $O(F_{\text{lower}}(f_1, f_2)) = O(f_1)+O(f_2)+O(D(f_1))+O(D(f_2)) = O(f_1)+O(f_2)$. The same logic yields $O(F_{\text{upper}}(f_1, f_2)) = O(f_1) + O(f_2)$. □

Appendix 2:

Proof of Inequation (7). We define

$$H_1 = D(f_1) - (f_2 - f_1)\sum_{i=1}^{n}\mu_i, \qquad H_2 = D(f_1) - (f_2 - f_1)\sum_{i=1}^{n}\lambda_i,$$

$$H_3 = D(f_2) - (f_2 - f_1)\sum_{i=1}^{n}\lambda_i, \qquad H_4 = D(f_2) - (f_2 - f_1)\sum_{i=1}^{n}\mu_i,$$

$$\Delta = (f_2 - f_1)\left(\sum_{i=1}^{n}\lambda_i + \sum_{i=1}^{n}\mu_i\right).$$

We have following four cases for H_1, H_2, H_3, and H_4.

Case 1. $H_1 \le H_3, H_2 \le H_4$ Case 2. $H_1 \le H_3, H_4 \le H_2$
Case 3. $H_3 \le H_1, H_2 \le H_4$ Case 4. $H_3 \le H_1, H_4 \le H_2$

For Case 1, $\text{Max}(H_1, H_3) = H_3$ and $\text{Min}(H_2, H_4) = H_2$. Therefore, $\text{Min}(H_2, H_4) - \text{Max}(H_1, H_3) = H_2 - H_3$. This and the fact that $H_2 - H_3 \le H_4 - H_3$ leads to

$$\text{Min}(H_2, H_4) - \text{Max}(H_1, H_3) = H_2 - H_3 \le H_4 - H_3 = \Delta$$

Almost the same logic leads to $\text{Min}(H_2, H_4) - \text{Max}(H_1, H_3) = \Delta$ for Cases 2, 3, and 4. Hence, Inequation (7) is true. □

Appendix 3:

Explanation of Carlier's method.

This appendix is a brief explanation of Carlier's method [7]. For a state vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$, we define $C_1(x)$ by $C_1(\boldsymbol{x}) = \{i | x_i = 1\}$, $C_0(x)$ by $C_0(\boldsymbol{x}) = \{i | x_i = 0\}$, and P by $P = \{\boldsymbol{x} \in \{0,1\}^n | \Psi(\boldsymbol{x}) = 1\}$, where P is the set of all path vectors.

The stationary availability A is computed as

$$A = \sum_{s \in P} \left\{ \prod_{j \in C_0(\boldsymbol{x})} (1 - A_j) \prod_{j \in C_1(\boldsymbol{x})} A_j \right\} \tag{A.7}$$

Carlier [7] assumes that each component of system S as a high level of availability, and this assumption results in that the product of $(1 - A_j)$ in (A.7) is small when the number of elements in $C_0(\boldsymbol{x})$ is big. For these cases, we can approximate A by ignoring the product of $(1 - A_j)$ and A_j in $\{\}$ in (A.7). The path vectors are obtained as follows.

(1) Enumerate $\boldsymbol{x}$ with $C_0(\boldsymbol{x})$ having zero elements, having one element, having two elements, etc., sequentially.

(2) If $\boldsymbol{x}$ in (1) satisfies $\Psi(\boldsymbol{x}) = 1$, then $\boldsymbol{x}$ is a path vector.

In the examples of Section 5, $\Psi(\boldsymbol{x})$ is determined by whether or not 12 paths are up in $\boldsymbol{x}$ in Figure 2, and 17 paths are up in Figure 3. The product of $(1 - A_j)$ and A_j in $\{\}$ in (A.7) is ignored if $C_0(\boldsymbol{x})$ has more than two elements for Figure 2 and more than three elements for Figure 3.

Masahiro Hayashi
Department of Information and Communication Engineering,
Faculty of Knowledge Engineering,
Tokyo City University,
1-28-1 Tamazutsumi, Setagaya-ku, Tokyo
158-8557, Japan
E-mail : mhaya@tcu.ac.jp