J Jpn Ind Manage Assoc 56, 439-446, 2006

Original

Recursive Formulas for the Reliability of Multi-State Consecutive-*k*-out-of-*n*:F System

Tomoaki Akiba
Information Management Engineering, Yamagata College of Industry and Technology, Yamagata, Yamagata, Japan
Hideki Nagatsuka[†]
Faculty of System Design, Tokyo Metropolitan University, Hino, Tokyo, Japan
Hisashi Yamamoto
Faculty of System Design, Tokyo Metropolitan University, Hino, Tokyo, Japan
and
Ming J. Zuo
Department of Mechanical Engineering , University of Alberta, Edmonton, Alberta, Canada

ABSTRACT
In traditional reliability theory, both the system and its components are allowed to take only two possible states: working or failed. In a multi-state system, both the system and the components are allowed to be in M+1 states: 0, 1, 2, ... , M, where M is a positive integer which represents a system or unit in perfect functioning state, while zero is complete failure state. A multi-state system reliability model provides more flexibility for the modeling of equipment conditions. Huang *et al.* (2003) proposed more general definitions of the multi-state consecutive-*k*-out-of-*n*:F and G systems and then provide an exact algorithm for evaluating the system state distribution of decreasing multi-state consecutive-*k*-out-of-*n*:F systems. Another algorithm is provided to bound the system state distribution of increasing multi-state consecutive-*k*-out-of-*n*:F and G systems. The multi-state consecutive-*k*-out-of-*n*:F system is applicable to, for example, quality control problems.

In this paper, we provide two theorems and a recursive algorithm which evaluate the system state distribution of a multi-state consecutive-*k*-out-of-*n*:F system using the theorems. These recursive formulas are useful for any multi-state consecutive-*k*-out-of-*n*:F system, including the decreasing multi-state F system, the increasing multi-state F system and other non-monotonic F systems. We calculate the order of computing time and memory capacity of the proposed algorithm and show that, in cases when the number of components n is large, the proposed algorithm is more efficient than other algorithms. A numerical experiment shows that when n is large, the proposed method is efficient for evaluating the system state distribution of multi-state consecutive-*k*-out-of-*n*:F systems.

Key words: Multi-state, consecutive-*k*-out-of-*n*:F system, system state distribution, recursive algorithm

1. INTRODUCTION

In traditional reliability theory, both the system and its components are allowed to take only two possible states: working or failed. In a multi-state system, both the system and its components may experience more than two possible states, for example, completely working, partially working, and completely failed. A multi-state system reliability model provides more flexibility for the modeling of equipment conditions.

In the binary context, a system with n components in sequence is called a consecutive-*k*-out-of-*n*:F (G) system if the system fails (works) whenever at least k consecutive components in the system fail (work). A consecutive-*n*-out-of-*n*:F (G) system is a parallel (series) system. Many research results have reported the reliability evaluation of binary consecutive-*k*-out-of-*n*:F and G systems; for example, see Chao *et al.*[1], Chiang and Niu[2], Hwang[5] and Kuo *et al.*[8].

In a multi-state system, both the system and the components are allowed to be in M+1 possible states, $0,1,2,\cdots,M$, where M is a positive integer which represents a system or unit in perfect functioning state, while zero is complete failure state.

Recently, researchers have extended the definitions of the binary consecutive-*k*-out-of-*n* system to the multi-state cases, for example, see Kossow and Preuss[6], Malinowski and Preuss ([9],[10]), Zuo and Liang[12], Koutras[7], and Haim and Porat[3].

Huang *et al.*[4] propose more general definitions of the multi-state consecutive-*k*-out-of-*n*:F and G systems, which is reviewed in the following section, and then provided an exact algorithm for evaluating the system state distribution of decreasing multi-state consecutive-*k*-out-of-*n*:F systems. Another algorithm is provided to bound the system state distribution of increasing multi-state consecutive-*k*-out-of-*n*:F and G systems. Zuo *et al.*[13] evaluated the system state distribution of decreasing multi-state consecutive-*k*-out-of-*n*:G systems. With minimal path vectors, they

[†] Phone: +81-42-585-8469
E-mail: hnagatsu@cc.tmit.ac.jp
Received: September 7, 2005
Accepted: December 16, 2005

provide a recursive formula is provided for evaluating the exact system state distribution when $M=3$. When $M \geq 4$, an algorithm is provided to bound the system state distribution. However, in the case of $M \geq 4$, an efficient method to evaluate the system state distribution has yet to be introduced.

In this paper, we propose new recursive formulas for evaluating the system state distribution of multi-state consecutive-*k*-out-of-*n*:F systems with $M \geq 1$ based on formulas provided by Yamamoto *et al.*[11]. These recursive formulas are useful for any multi-state consecutive-*k*-out-of-*n*:F system, including the decreasing multi-state F system, the increasing multi-state F system and other non-monotonic F systems. We also evaluate the orders of computation time and memory requirements of our proposed algorithm based on the recursive formulas. A numerical experiment shows that when *n* is large, the proposed method is efficient for evaluating the system state distribution of the multi-state consecutive-*k*-out-of-*n*:F systems.

2. THE MULTI-STATE CONSECUTIVE-*k*-OUT-OF-*n*:F SYSTEM

In this section, we review the definitions of the multi-state consecutive-*k*-out-of-*n*:F system proposed by Huang *et al.*[4]. Before definitions, we define notation as follows.

u_i : state of component *i*, $u_i \in \{0,1,\cdots,M\}$, for $i=1,2,\cdots,n$

$\mathbf{u}$: the vector of component states, $\mathbf{u}=(u_1,u_2,\cdots,u_n)$

$\varphi(\mathbf{u})$: the system structure function representing the state of the system, $\varphi(\mathbf{u}) \in \{0,1,\cdots,M\}$

M+1 : the number of the system state of the multi-state consecutive-*k*-out-of-*n*:F system or its components are allowed to be in *M*+1 possible states, $0,1,2,\cdots,M$; where, for example, *M* is a positive integer which represents a system or unit in perfect functioning state, while zero is complete failure state.

Definition (Huang *et al.*[4])
$\varphi(\mathbf{u})<j$ ($j=1,2,\cdots,M$) if at least k_l consecutive components are in states below *l* for all *l* such that $j \leq l \leq M$. An *n*-component system with such a property is called a multi-state consecutive-*k*-out-of-*n*:F system.

The condition in this definition can also be phrased as follows: $\varphi(\mathbf{u})<j$ ($j=1,2,\cdots,M$) if at least k_j consecutive components are in states below *j*, at least k_{j+1} consecutive components are in states below *j*+1, ..., and at least k_M consecutive components are in states below *M*.

Note that the multi-state consecutive-*k*-out-of-*n*:F system becomes a consecutive-*k*-out-of-*n*:F system when $M=1$ and $k_1=k$.

The multi-state consecutive-*k*-out-of-*n*:F system is called a decreasing (increasing) multi-state consecutive-*k*-out-of-*n*:F system when $k_0(\equiv n) \geq k_1 \geq k_2 \geq \cdots \geq k_M$ ($k_0(\equiv 0) \leq k_1 \leq k_2 \leq \cdots \leq k_M$).

The multi-state consecutive-*k*-out-of-*n*:F system is applicable to, for example, quality control problems.

Example (Huang *et al.*[4])
A batch of products may be sorted into one of the following three classes based on the level of quality: grade A, grade B, and rejected. The following sampling procedure is used to classify the product items: if consecutive-3-out-of-10 items of a sample do not meet the standard of grade A, then a subsequent inspection is conducted under the standard of grade B; otherwise, it is labeled grade A. If consecutive 5-out-of-10 items of a sample are judged to be lower than grade B, then this batch will be rejected; otherwise, it is labeled grade B. For such a problem, we can define a multi-state consecutive- *k*-out-of-*n*:F system with the label of the batch as the system state and the sampled items as components. Both the system and components have three possible states: state 2 (grade A), state 1 (grade B), and state 0 (rejected). At system state level 2, it has a consecutive-3-out-of-10:F structure, and the system state level 1 it has a consecutive-5-out-of-10:F structure.

Throughout this paper, we assume the states of components are mutually statistically independent.

3. THEOREMS AND ALGORITHM

In this section, we provide theorems for evaluating the system state distribution of the multi-state consecutive-*k*-out-of-*n*:F system. Let *n* be the number of components. We denote the probability the state of the multi-state consecutive-*k*-out-of-*i*:F system is below *j* by $R^{(j)}(i)$, for $i=1,\cdots,n$, $j=1,2,\cdots,M$. Let $\mathbf{x}=(x_j,x_{j+1},\cdots,x_M)$ denote the $(M-j+1)$ -dimensional vector which is explained below in detail, where $x_l=0,1,2,\cdots,k_l,\bar{k}_l,k_l+1$ ($l=j,j+1,\cdots,M$), for $j=1,2,\cdots,M$, where $\bar{k}_l$ and k_l+1 are not numbers but symbolic characters. For $i=1,\cdots,n$, $j=1,2,\cdots,M$, $x_l=0,1,\cdots,k_l,\bar{k}_l,k_l+1$ ($l=j,j+1,\cdots,M$), we define the event $S(i,l;x_l)$ that

1) "k_l consecutive components with state below l do not occur from component 1 to $i-1$" and "the state of component i is in some state l or above" if $x_l = 0$ and $i = 2,\cdots,n$; "the state of component 1 is in some state l or above" if $x_l = 0$ and $i = 1$, and
2) "k_l consecutive components with state below l do not occur from component 1 to $i - x_l - 1$" and "x_l components from component $i - x_l + 1$ to i are in a state below l" and "the state of component $i - x_l$ is in some state l or above" if $x_l = 1,2,\cdots,k_l$ and $x_l \le i-2$;
"x_l components from component $i - x_l + 1$ to i are in a state below l" and "the state of component $i - x_l$ is in some state l or above" if $x_l = 1,2,\cdots,k_l$ and $x_l \ge i-1$, and
3) "at least k_l consecutive components with a state below l occur from component 1 to $i-1$" and "the state of component i is in any state" if $x_l = \bar{k}_l$ and $i = 2,\cdots,n$, and
null event if $x_l = \bar{k}_l$ and $i = 1$, and
4) "at least k_l consecutive components with a state below l occur from component 1 to i" if $x_l = k_l + 1$.

In the above definition of $S(i,l;x_l)$, for $i\ (i \le 0)$, we suppose that such hypothetical component i exists and its states are always M. As well note that $S(i,l;k_l+1) = S(i,l;k_l) \cup S(i,l;\bar{k}_l)$, $S(i,l;k_l) \cap S(i,l;\bar{k}_l)$ is null and $S(i,l;\bar{k}_l) = S(i-1,l;k_l+1)$.

Let $R^{(j)}(i;\mathbf{x}) = \Pr\left\{\bigcap_{l=j}^{M} S(i,l,x_l)\right\}$ if $i = 1,\cdots,n$, and

$$R^{(j)}(0;\mathbf{x}) \equiv \begin{cases} 1 & \text{if } \mathbf{x} = (0,0,\cdots,0), \\ 0 & \text{if } \mathbf{x} \ne (0,0,\cdots,0). \end{cases}$$

By considering the relation between $R^{(j)}(i-1;\mathbf{x})$ and $R^{(j)}(i;\mathbf{x})$ carefully, we obtain recursive formulas (Theorem 1). Theorem 1 enables us to evaluate the system state distribution of a multi-state consecutive-k-out-of-n:F system efficiently. Before showing Theorem 1, we define the notations as follows.

$A(\mathbf{x})$	:	$\{l \mid x_l = 1,2,\cdots,k_l (l = j,\cdots,M)\}$; that is, $\{l \mid x_l \ne 0, \bar{k}_l\ (l = j,\cdots,M)\}$, for $x_l = 0,1,2,\cdots,k_l,\bar{k}_l\ (l = j,\cdots,M)$ and $j = 1,\cdots,M$
$B(\mathbf{x})$	:	$\{l \mid x_l = 0 (l = j,\cdots,M)\}$, for $x_l = 0,1,2,\cdots,k_l,\bar{k}_l\ (l = j,\cdots,M)$ and $j = 1,2,\cdots,M$
$a(\mathbf{x})$	:	$\min\{l \in A(\mathbf{x})\}$, for $A(\mathbf{x}) \ne \phi$
$b(\mathbf{x})$	:	$\max\{l \in B(\mathbf{x})\}$, for $B(\mathbf{x}) \ne \phi$

For example, we assume $k_1 = 3$, $k_2 = 2$, $k_3 = 2$, $k_4 = 3$, and $k_5 = 3$. If $\mathbf{x} = (0,2,\bar{2},2,4)$, then $A(\mathbf{x}) = \{2,4\}$, $a(\mathbf{x}) = 2$, $B(\mathbf{x}) = \{1\}$ and $b(\mathbf{x}) = 1$. Also, if $\mathbf{x} = (0,0,\bar{2},2,\bar{3})$, then $A(\mathbf{x}) = \{4\}$, $a(\mathbf{x}) = 4$, $B(\mathbf{x}) = \{1,2\}$ and $b(\mathbf{x}) = 2$.

Furthermore, we define the notations as follows.

p_{ij}	:	probability that component i is in state j, for $i = 1,\cdots,n$ and $j = 1,\cdots,M$
P_{ij}	:	$\sum_{l=j}^{M} p_{il}$, for $i = 1,\cdots,n$ and $j = 1,\cdots,M$
Q_{ij}	:	$\sum_{l=0}^{j-1} p_{il}$, for $i = 1,\cdots,n$ and $j = 1,\cdots,M$

We are now ready to present Theorem 1.

THEOREM 1
Suppose that the states of components of the system are mutually statistically independent. Let $F(i;\mathbf{x})$ be the probability that component i is in a state which makes the event $\bigcap_{l=j}^{M} S(i,l;x_l)$ occur, for $x_l = 0,1,2,\cdots,k_l,\bar{k}_l\ (l = j, j+1,\cdots,M)$, $i = 1,2,\cdots,n$ and $j = 1,2,\cdots,M$. Then,

1) For $i = 1,2,\cdots,n$,

$$R^{(j)}(i) = R^{(j)}(i;k_j+1, k_{j+1}+1,\cdots,k_M+1). \tag{1}$$

2) For $i = 1,2,\cdots,n$, if $x_l = k_l + 1$,

$$R^{(j)}(i;x_j,\cdots,x_l,\cdots,x_M) = R^{(j)}(i;x_j,\cdots,k_l,\cdots,x_M) + R^{(j)}(i;x_j,\cdots,\bar{k}_l,\cdots,x_M). \tag{2}$$

3) For $i = 1,2,\cdots,n$ and $x_l = 0,1,\cdots,k_l,\bar{k}_l$ $(l = j, j+1,\cdots M)$,

$$R^{(j)}(i;\mathbf{x}) = \begin{cases} 0, & \text{if } x_{l_1} > x_{l_2}\ (l_1 < l_2) \\ & \quad \text{for } l_1, l_2 \in A(\mathbf{x}) \cup B(\mathbf{x}) \\ & \text{or if there exsits } x_l \text{ such that } x_l > i \\ & \quad \text{for } l \in A(\mathbf{x}) \cup B(\mathbf{x}), \\ F(i;\mathbf{x}) \sum_{\mathbf{y}\in\Omega(\mathbf{x})} R^{(j)}(i-1;\mathbf{y}), & \text{otherwise,} \end{cases} \tag{3}$$

where

$$F(i;\mathbf{x}) = \begin{cases} 1, & \text{if } A(\mathbf{x}) = \phi \text{ and } B(\mathbf{x}) = \phi, \\ Q_{i,a(\mathbf{x})}, & \text{if } A(\mathbf{x}) \neq \phi \text{ and } B(\mathbf{x}) = \phi, \\ P_{i,b(\mathbf{x})}, & \text{if } A(\mathbf{x}) = \phi \text{ and } B(\mathbf{x}) \neq \phi, \\ P_{i,b(\mathbf{x})} - P_{i,a(\mathbf{x})}, & \\ & \text{if } A(\mathbf{x}) \neq \phi \text{ and } B(\mathbf{x}) \neq \phi, \end{cases} \tag{4}$$

and

$$\mathbf{y} = (y_j, y_{j+1}, \cdots, y_M),$$

$$\begin{aligned} \Omega(\mathbf{x}) = \{\mathbf{y} \mid\, & y_l = k_l + 1, && \text{if } x_l = \bar{k}_l, \\ & y_l = x_l - 1, && \text{if } x_l = 1, \cdots, k_l, \\ & y_l = 0, 1, \cdots, k_l - 1, && \text{if } x_l = 0 \quad \}. \end{aligned} \tag{5}$$

As the boundary condition, for $i = 0$,

$$R^{(j)}(i;\mathbf{x}) \equiv \begin{cases} 1, & \text{if } \mathbf{x} = (0,0,\cdots 0), \\ 0, & \text{if } \mathbf{x} \neq (0,0,\cdots 0). \end{cases} \tag{6}$$

The proof of Theorem 1 is provided in the appendix. From Theorem 1, we obtain $R^{(j)}(n)$ when we want to get only the value of this probability. However, using Theorem 2, we obtain $R^{(j)}(n)$ for $j = 2, \cdots, M$ with less effort when obtaining $R^{(1)}(n)$.

THEOREM 2

For $j = 1, 2, \cdots, M$ and $i = 1, 2, \cdots, n$,

$$R^{(j)}(i) = \sum_{l_1 \in C_1} \cdots \sum_{l_{j-1} \in C_{j-1}} R^{(1)}(i; l_1, \cdots, l_{j-1}, k_j + 1, \cdots, k_M + 1), \tag{7}$$

where $C_t = \{0, 1, \cdots, k_t - 1, k_t + 1\}$, $t = 1, \cdots, j - 1$.

Proof: It is easy to see that the event, whose probability is equal to each term on the right-hand side of Eq.(7) is equivalent to the event that at least k_j consecutive components to be in states below j occur; ... ; at least k_M consecutive components to be in states below M occur. This corresponds to the event that the state of a multi-state consecutive-k-out-of-i:F system is j or above from the definition of the system. Thus, Eq.(7) holds.

We demonstrate how to obtain $R^{(1)}(n)$ and $R^{(2)}(n)$ in the case of M=2 using theorems 1 and 2.

Example (the case of $M = 2$)

From Eq.(1),

$R^{(1)}(i) = R^{(1)}(i; k_1 + 1, k_2 + 1)$,

for $i = 1, 2, \cdots, n$.

From Eq.(2), for $x_1 = k_1 + 1$ and/or $x_2 = k_2 + 1$,

$$R^{(1)}(i;(x_1,x_2)) = \begin{cases} R^{(1)}(i;(k_1,x_2)) + R^{(1)}(i;(\bar{k}_1,x_2)), & \\ \qquad \text{if } x_1 = k_1 + 1 \text{ and } x_2 \neq k_2 + 1, & \\ R^{(1)}(i;(x_1,k_2)) + R^{(1)}(i;(x_1,\bar{k}_2)), & \\ \qquad \text{if } x_1 \neq k_1 + 1 \text{ and } x_2 = k_2 + 1, & \\ R^{(1)}(i;(k_1,k_2)) + R^{(1)}(i;(\bar{k}_1,k_2)) & \\ \quad + R^{(1)}(i;(k_1,\bar{k}_2)) + R^{(1)}(i;(\bar{k}_1,\bar{k}_2)), & \\ \qquad \text{if } x_1 = k_1 + 1 \text{ and } x_2 = k_2 + 1, & \end{cases}$$

for $i = 1, 2, \cdots, n$.

From Eq.(3), for $x_1 = 0, 1, \cdots, k_1, \bar{k}_1$ and $x_2 = 0, 1, \cdots, k_2, \bar{k}_2$,

$$R^{(1)}(i;(x_1,x_2)) = \begin{cases} R^{(1)}(i-1;(k_1+1,k_2+1)), & \\ \qquad \text{if } x_1 = \bar{k}_1 \text{ and } x_2 = \bar{k}_2, & \\ Q_{i1} R^{(1)}(i-1;(x_1-1,k_2+1)), & \\ \qquad \text{if } 1 \leq x_1 \leq k_1 \text{ and } x_2 = \bar{k}_2, & \\ P_{i1} \sum_{t=0}^{k_1-1} R^{(1)}(i-1;(t,k_2+1)), & \\ \qquad \text{if } x_1 = 0 \text{ and } x_2 = \bar{k}_2, & \\ Q_{i2} R^{(1)}(i-1;(k_1+1,x_2-1)), & \\ \qquad \text{if } x_1 = \bar{k}_1 \text{ and } 1 \leq x_2 \leq k_2, & \\ Q_{i1} R^{(1)}(i-1;(x_1-1,x_2-1)), & \\ \qquad \text{if } 1 \leq x_1 \leq k_1 \text{ and } 1 \leq x_2 \leq k_2, x_1 \leq x_2, & \\ 0, & \text{if } 1 \leq x_1 \leq k_1 \text{ and } 1 \leq x_2 \leq k_2, x_1 > x_2, \\ 0, & \text{if } 1 \leq x_1 \leq k_1 \text{ and } x_2 = 0, \\ P_{i2} \sum_{t=0}^{k_2-1} R^{(1)}(i-1;(k_1+1,t)), & \\ \qquad \text{if } x_1 = \bar{k}_1 \text{ and } x_2 = 0, & \\ p_{i1} \sum_{t=0}^{k_1-1} R^{(1)}(i-1;(t,x_2-1)), & \\ \qquad \text{if } x_1 = 0 \text{ and } 1 \leq x_2 \leq k_2, & \\ P_{i2} \sum_{t_1=0}^{k_1-1} \sum_{t_2=0}^{k_2-1} R^{(1)}(i-1;(t_1,t_2)), & \\ \qquad \text{if } x_1 = 0 \text{ and } x_2 = 0. & \end{cases}$$

As the boundary condition, for $i = 0$,

$$R^{(1)}(i;(x_1,x_2)) \equiv \begin{cases} 1, & \text{if } (x_1,x_2) = (0,0), \\ 0, & \text{if } (x_1,x_2) \neq (0,0). \end{cases}$$

Furthermore, from Eq.(7),

$$R^{(2)}(n) = \sum_{l_1 \in \{0,1,\ldots,k_1-1,k_1+1\}} R^{(1)}(n; l_1, k_2 + 1) \cdot$$

Using theorems 1 and 2, the proposed algorithm consists of the following steps, for computing $R^{(j)}(n)$'s for $j=1,2,\cdots,M$.

STEP 0 (Setting initial value)
Set $i=0$ and

$$R^{(j)}(i;\mathbf{x})=\begin{cases}1, & \text{if } \mathbf{x}=(0,0,\cdots,0),\\ 0, & \text{if } \mathbf{x}\neq(0,0,\cdots,0).\end{cases}$$

STEP 1
Set $i=i+1$.
Obtain $R^{(l)}(i;\mathbf{x})$s for all $\mathbf{x}$ such that x_l does not take k_l+1 for all l, by Eqs.(3), (4) and (5). Go to Step 2

STEP 2
Obtain $R^{(l)}(i;\mathbf{x})$ for all $\mathbf{x}$ such that x_l takes k_l+1 for some l, by Eq.(2). Go to Step 3.

STEP 3
Go to Step1 if $i<n$, and go to Step 4 if $i=n$.

STEP 4
Obtain the value of $R^{(j)}(n)$'s for $j=2,\cdots,M$ from Eqs.(1) and (7).

4. THE ORDER OF OUR ALGORITHM

We have evaluated the orders of computation time and memory size for the proposed algorithm when theorems 1 and 2 are used. For each *i*, in order to compute $R^{(l)}(i;\mathbf{x})$ for $x_l=k_l+1$, we must use Eq.(2) a maximum of 2^M times. The number of $R^{(j)}(i;\mathbf{x})$'s for $x_l=0,1,\cdots,k_l,\bar{k}_l$ $(l=j,j+1,\cdots,M)$ is $\prod_{l=j}^{M}(k_l+2)$. Therefore, the order of computing $R^{(j)}(n)$'s for $j=1,2,\cdots,M$ is $O\left(n(2^M+\prod_{l=j}^{M}k_l)\right)$.
The maximum memory size required for computing $R^{(j)}(i;\mathbf{x})$ is $2\prod_{l=j}^{M}(k_l+3)$, because we need to have $\prod_{l=j}^{M}(k_l+3)$ entries for $i-1$ and i at the same time. Therefore, the order of the required memory size is $O\left(\prod_{l=j}^{M}k_l\right)$. The order of computation time is of exponential of M and polynomial of n. The required memory size is also of an exponential order of M, but does not depend on n.

5. NUMERICAL EXPERIMENTS

We performed a numerical experiment in order to compare the proposed algorithm with other algorithms. All the experiments were executed using a Pentium-M (1.3GHz) computer with 768MBytes of RAM, MS-Windows 2000, Visual C++.NET and C language programming. For the first numerical experiments, we consider the following three systems.

1) The three-state consecutive-*k*-out-of-*n*:F system with $k_1=3$, $k_2=2$ and $k_3=1$ including i.i.d. components. The state distribution of each component is $p_{i0}=0.1$, $p_{i1}=0.2$, $p_{i2}=0.3$ and $p_{i3}=0.4$,
2) Five-state consecutive-*k*-out-of-*n*:F system with $k_1=6$, $k_2=5$, $k_3=4$, $k_4=3$ and $k_5=2$ including i.i.d. components. The state distribution of each component is $p_{i0}=0.05$, $p_{i1}=0.05$, $p_{i2}=0.1$, $p_{i3}=0.1$, $p_{i4}=0.2$ and $p_{i5}=0.5$.
3) The six-state consecutive-*k*-out-of-*n*:F system with $k_1=6$, $k_2=5$, $k_3=4$, $k_4=3$, $k_5=2$ and $k_6=1$ including i.i.d. components. The state distribution of each component is $p_{i0}=0.05$, $p_{i1}=0.05$, $p_{i2}=0.1$, $p_{i3}=0.1$, $p_{i4}=0.2$, $p_{i5}=0.2$ and $p_{i6}=0.3$.

The number *n* of components and the number *M* of states are varied and the computation times are compared between the proposed algorithm and enumeration method, as shown in Table 1. The averages are the results from five trials for each *n* value. In Table 1, we marked "N/A" when it took over 1hr. Clearly, the proposed algorithm is faster than the enumeration method, especially, when the number of components *n* is large. Additionally, the proposed algorithm requires more computation time for *M*, but the computation time is very small even if $M=6$. Table 2 shows the system state distribution of the six-state consecutive-*k*-out-of-*n*:F system obtained using the proposed algorithm.

From these results, we see that the proposed algorithm is very efficient for evaluating the system

Table 1: Comparison of the computation time

M	n	$R^{(M)}(n)$	Average computation time (sec.)	
			Proposed algorithm	Enumeration method
3	8	0.999345	0.01	0.12
	12	0.999983	0.01	41.63
	15	0.999999	0.01	1836.07
	20	1.000000	0.01	N/A
	100	1.000000	0.03	N/A
5	8	0.785156	0.01	2.47
	12	0.907959	0.01	N/A
	15	0.951263	0.01	N/A
	20	0.983109	0.01	N/A
	100	1.000000	0.05	N/A
6	8	0.999934	0.01	9.54
	10	0.999994	0.04	N/A
	15	1.000000	0.04	N/A
	20	1.000000	0.05	N/A
	100	1.000000	0.46	N/A

Table2: System state distribution (M=6)

n	$R^{(2)}(n)$	$R^{(3)}(n)$	$R^{(4)}(n)$	$R^{(5)}(n)$	$R^{(6)}(n)$	Computation time (sec.)
6	0.000019	0.004160	0.083700	0.671875	0.999271	0.00
7	0.000028	0.005440	0.102090	0.734375	0.999781	0.02
8	0.000037	0.006720	0.120122	0.785156	0.999934	0.03
9	0.000046	0.007998	0.137797	0.826172	0.999980	0.03
10	0.000055	0.009274	0.155116	0.859375	0.999994	0.04
20	0.000145	0.021947	0.310275	0.983109	1.000000	0.05
30	0.000235	0.034459	0.436940	0.997971	1.000000	0.06
40	0.000325	0.046810	0.540344	0.999756	1.000000	0.08
50	0.000415	0.059003	0.624758	0.999971	1.000000	0.12
60	0.000505	0.071040	0.693669	0.999996	1.000000	0.15
70	0.000595	0.082923	0.749926	1.000000	1.000000	0.20
80	0.000685	0.094654	0.795851	1.000000	1.000000	0.27
90	0.000775	0.106235	0.833342	1.000000	1.000000	0.36
100	0.000865	0.117668	0.863948	1.000000	1.000000	0.46

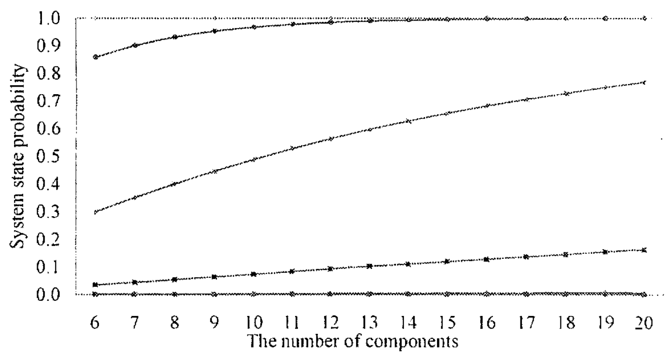

Figure 1 The system state distribution with $k_1 = 6, k_2 = 5, k_3 = 4, k_4 = 3, k_5 = 2$ and $k_6 = 1$

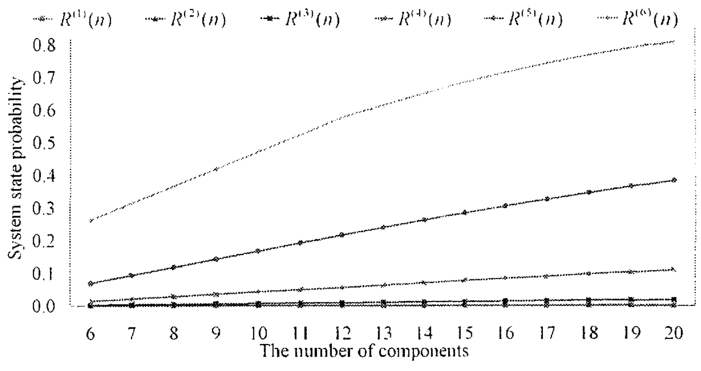

Figure 2 The system state distribution with $k_1 = k_2 = k_3 = k_4 = k_5 = k_6 = 6$

state distribution and enables the system state distribution in the case of large n values to be calculated.

In Fig.1, we illustrate the system state distribution of the six-state consecutive-k-out-of-n:F system with $k_1 = 6$, $k_2 = 5$, $k_3 = 4$, $k_4 = 3$, $k_5 = 2$ and $k_6 = 1$, where the state distribution of each component is $p_{i0} = 0.1$, $p_{i1} = 0.12$, $p_{i2} = 0.13$, $p_{i3} = 0.14$, $p_{i4} = 0.15$, $p_{i5} = 0.16$ and $p_{i6} = 0.2$. In Fig.2, we illustrate the system state distribution of the six-state consecutive-k-out-of-n:F system with $k_l = 6$ for $l = 1,2,\cdots,6$, where the state distribution of each component is the same as Fig.1. As k_l in Fig.1 is larger than or equal to k_l in Fig.2, the system state distribution shown in Fig. 2 is larger than one in Fig.1.

6. CONCLUSION

In this paper, we proposed a new recursive algorithm for the system state distribution of multi-state consecutive-k-out-of-n:F systems. We evaluated the proposed algorithm in terms of the orders of computation time and memory size requirements. Numerical experiments showed that the proposed algorithm is very effective for evaluating the system state distribution of the multi-state consecutive-k-out-of-n:F systems when n is large.

APPENDIX

Proof of Theorem 1:

Eqs.(1) and (2) can be directly proven from the definition of $R^{(j)}(i;\mathbf{x})$.

Eq.(3) can be proven in the following.

1) In the cases of $x_{l_1} > x_{l_2}$ $(l_1 < l_2)$ for $l_1, l_2 \in A(\mathbf{x}) \cup B(\mathbf{x})$

From the definition of $R^{(j)}(i;\mathbf{x})$ and the considered system, $R^{(j)}(i;\mathbf{x}) = 0$ as " x_{l_1} consecutive components to be in a state below l_1 " means " x_{l_1} consecutive components to be in state l_2 ."

2) In the cases of existing some x_l such that $x_l > i$

for $l \in A(\mathbf{x}) \bigcup B(\mathbf{x})$

$R^{(j)}(i;\mathbf{x}) = 0$ because $S(i,l;x_l)$ for $x_l > i$ is null.

3) Other cases

From the definition of $S(i,l;x_l)$, we can verify the following relations. For $i = 1,2,\cdots,n$, let Z_i be the random variable that takes l when the state of component i is l, where $l = 0,1,2,\cdots,M$. For $x_l = 0,1,\cdots,k_l,\bar{k}_l$ $(l = j, j+1,\cdots,M)$ and $i = 1,2,\cdots,n$ and $j = 1,2,\cdots,M$,

$$S(i,l;x_l) = \begin{cases} S(i-1,l;k_l+1), & \text{if } x_l = \bar{k}_l, \\ S(i-1,l;x_l-1) \cap \{\mathbf{Z}_i < l\}, & \text{if } x_l = 1,2,\cdots,k_l, \\ \left\{ \bigcup_{y_l=0}^{k_l-1} S(i-1,l;y_l) \right\} \cap \{\mathbf{Z}_i \geq l\}, & \text{if } x_l = 0, \end{cases}$$

where

$$S(i,l;x_l) \equiv \begin{cases} \text{null event}, & \text{if } x_l = 1,2,\cdots,k_l,\bar{k}_l, \\ \text{whole event}, & \text{if } x_l = 0, \end{cases}$$

for $i = 0$, for convenience.

From this, for $x_l = 0,1,\cdots,k_l,\bar{k}_l$ $(l = j, j+1,\cdots,M)$ and $i = 1,2,\cdots,n$, $S(i,l;x_l)$ can be expressed by

$$S(i,l;x_l) = \bigcup_{y_l \in \Omega_l(x_l)} S(i-1,l;y_l) \cap C_{il}(x_l),$$

where $C_{il}(x_l)$ means a whole event if $x_l = \bar{k}$ and

$$C_{il}(x_l) = \begin{cases} \{\mathbf{Z}_i < l\}, & \text{if } x_l = 1,2,\cdots,k_l, \\ \{\mathbf{Z}_i \geq l\}, & \text{if } x_l = 0, \end{cases}$$

and $\Omega_l(x_l)$ means the set defined by

$$\begin{aligned} \Omega_l(x_l) = \{y_l \mid\ & y_l = k_l + 1, && \text{if } x_l = \bar{k}_l, \\ & y_l = x_l - 1, && \text{if } x_l = 1,\cdots,k_l, \\ & y_l = 0,1,\cdots,k_l - 1, && \text{if } x_l = 0 \quad \}. \end{aligned}$$

Therefore, we get

$$\begin{aligned} & R^{(j)}(i;\mathbf{x}) \\ & = \Pr\left\{ \bigcap_{l=j}^{M} S(i,l;x_l) \right\} \\ & = \Pr\left\{ \bigcap_{l=j}^{M} \left\{ \bigcup_{y_l \in \Omega_l(x_l)} S(i-1,l;y_l) \cap C_{il}(x_l) \right\} \right\} \\ & = \Pr\left\{ \left\{ \bigcap_{l=j}^{M} \left\{ \bigcup_{y_l \in \Omega_l(x_l)} S(i-1,l;y_l) \right\} \right\} \cap \left\{ \bigcap_{l=j}^{M} C_{il}(x_l) \right\} \right\} \\ & = \Pr\left\{ \left\{ \bigcup_{\substack{y_l \in \Omega_l(x_l) \\ l=1,2,\cdots,j}} \bigcap_{l=j}^{M} S(i-1,l;y_l) \right\} \cap \left\{ \bigcap_{l=j}^{M} C_{il}(x_l) \right\} \right\} \\ & = \Pr\left\{ \bigcup_{\mathbf{y} \in \Omega(\mathbf{x})} S(i-1,l;y_l) \right\} \Pr\left\{ \bigcap_{l=j}^{M} C_{il}(x_l) \right\}, \end{aligned} \tag{A1}$$

where $\Omega(\mathbf{x})$ is given by Eq.(5). The last equality holds from the independence of component states.

Next, we let $F(i;\mathbf{x}) \equiv \Pr\left\{ \bigcap_{l=j}^{M} C_{il}(x_l) \right\}$ and show $F(i;\mathbf{x})$ can be given by Eq.(4) in the following.

When $A(\mathbf{x}) = \phi$ and $B(\mathbf{x}) = \phi$, all x_l must be $\bar{k}_l$, that is, $C_{il}(x_l) = \Omega$ for all l. Therefore, $F(i;\mathbf{x}) = 1$.

When $A(\mathbf{x}) \neq \phi$ and $B(\mathbf{x}) = \phi$, all x_l must not be 0. So, as $C_{il}(x_l) = \{\mathbf{Z}_i < l\}$ for all l except $x_l = \bar{k}_l$, $F(i;\mathbf{x}) = \Pr\{\mathbf{Z}_i < a(\mathbf{x})\} = Q_{i,a(\mathbf{x})}$.

When $A(\mathbf{x}) = \phi$ and $B(\mathbf{x}) \neq \phi$, all x_l must be 0 or $\bar{k}_l$. Therefore, as $C_{il}(x_l) = \{\mathbf{Z}_i \geq l\}$ for all l except $x_l = \bar{k}_l$, $F(i;\mathbf{x}) = \Pr\{\mathbf{Z}_i \geq b(\mathbf{x})\} = P_{i,b(\mathbf{x})}$.

When $A(\mathbf{x}) \neq \phi$ and $B(\mathbf{x}) \neq \phi$, $C_{il}(x_l) = \{\mathbf{Z}_i \geq l\}$ for $x_l = 0$ and $C_{il}(x_l) = \{\mathbf{Z}_i < l\}$ for $x_l \neq 0$ and $x_l \neq \bar{k}_l$. From case 1), $R^{(j)}(i;\mathbf{x}) = 0$ for $\mathbf{x}$ with $x_{l_1} > x_{l_2}$ $(l_1 < l_2)$ for $l_1, l_2 \in A(\mathbf{x}) \bigcup B(\mathbf{x})$. So, we may consider only the cases of $x_{l_1} \leq x_{l_2} \leq \cdots \leq x_{l_t}$ $(l_1 < l_2 < \cdots < l_t,\ l_i \in A(\mathbf{x}) \bigcup B(\mathbf{x}),\ i = 1,2,\cdots,t)$, where t means the number of elements of $A(\mathbf{x}) \bigcup B(\mathbf{x})$. It is easy to see $b(\mathbf{x}) < a(\mathbf{x})$ in these cases. Therefore,

$$\begin{aligned} F(i;\mathbf{x}) & = \Pr\{\{\mathbf{Z}_i < a(\mathbf{x})\} \cap \{\mathbf{Z}_i \geq b(\mathbf{x})\}\} \\ & = P_{i,b(\mathbf{x})} - P_{i,a(\mathbf{x})}. \end{aligned}$$

Therefore, Eq.(4) can be proven and Eq.(3) holds by noting $\Pr\left\{ \bigcup_{\mathbf{y} \in \Omega(\mathbf{x})} S(i-1,l;y_l) \right\} = \sum_{\mathbf{y} \in \Omega(\mathbf{x})} R^{(j)}(i-1;\mathbf{y})$.

Finally, Eq.(6) holds from the definition of $R^{(j)}(i;\mathbf{x})$.

Q.E.D

ACKNOWLEDGMENTS

The authors are grateful to the associate editor and anonymous referees for their comments and suggestions, which have led to the improvement of our study and this paper. This research was partially supported by the Natural Sciences and Engineering Research Council of Canada, and by Grant#17510129, Grant-in-Aid for Scientific Research(C) from JSPS,(2005-).

REFERENCES

[1] Chao, M. T., Fu, J. C., and Koutras, M. V.: "Survey of reliability studies of consecutive-*k*-out-of-*n*:F & related Systems," *IEEE Transactions on Reliability* , Vol.44, No.1, pp. 120-125 (1995)

[2] Chiang, D. T. and Niu, S.: "Reliability of consecutive-*k*-out-of-*n*:F system," *IEEE Transactions on Reliability*, Vol.30, No.1, pp. 87-89 (1981)

[3] Haim, M. and Porat, Z.: "Bayes reliability modeling of a multistate consecutive-*k*-out-of-*n*:F system," *Proceeding Annual Reliability and Maintainability Symposium*, pp. 582-586 (1991)

[4] Huang, J., Zuo, M. J., and Fang, Z.: "Multi-state consecutive *k*-out-of-*n* systems," *IIE Transactions Quality and Reliability Engineering*, Vol. 35, pp. 527-534 (2003)

[5] Hwang, F. K.: "Fast solutions for consecutive-*k*-out-of-*n*:F System," *IEEE Transactions on Reliability*, Vol. 31, No. 5, pp. 447-448 (1982)

[6] Kossow, A. and Preuss, W.: "Reliability of linear consecutively-connected systems with multistate Components," *IEEE Transactions on Reliability*, Vol. 44, No. 3, pp. 518-522 (1995)

[7] Koutras, M. V.: "Consecutive-*k*,*r*-out-of-*n*:DFM systems," *Microelectronics and Reliability*, Vol. 37, No. 4, pp.597-603 (1997)

[8] Kuo, W., Fu, J. C., and Lou, W. Y.: "Opinions on consecutive-*k*-out-of-*n*:F System," *IEEE Transactions on Reliability*, Vol. 43, No. 4, pp. 659–662 (1994)

[9] Malinowski, J. and Preuss, W.: "Reliability of circular consecutively-connected systems with multi-state components," *IEEE Transactions on Reliability*, Vol. 44, No. 3, pp.532-534 (1995)

[10] Malinowski, J. and Preuss, W.: "Reliability of reverse-Tree-Structured systems with multi-state components," *Microelectronics and Reliability*, Vol. 36, No. 1, pp.1-7 (1996)

[11] Yamamoto, H., Zuo, M.J. Akiba, T. and Tian, Z.: "Recursive Formulas for The Reliability of Multi-state Consecutive-*k*-out-of-*n*:G Systems," (to appear)

[12] Zuo, M. J. and Liang, M.: "Reliability of multistate consecutively-connected systems," *Reliability Engineering and System Safety*, Vol. 44, pp. 173-176 (1994)

[13] Zuo, M. J., Fang, Z., Huang, J., and Xu, X.: "Performance evaluation of decreasing multi-state consecutive-*k*-out-of-*n*:G systems," *International Journal of Reliability, Quality and Safety Engineering*, Vol. 10, No. 3, pp. 345-358 (2003)