New algorithm to reduce the number of computing steps in reliability formula of weighted–k–out–of–n system

Yoichi Higashiyama[1] **and Tatsunari Ohkura**[2]
Department of Electrical and Electronic Engineering
Ehime University
Matsuyama, 790-8577 Japan
[1] **e-mail: mountain@dpc.ehime–u.ac.jp**
[2] **e-mail: ohkura@akitsu.ee.ehime-u.ac.jp**

and

Ventsi Rumchev
Department of Mathematics and Statistics
Curtin University of Technology
P.O. Box U1987, Perth, WA 6845 Australia
e-mail: rumchevv@curtin.edu.au

Abstract

In the disjoint products version of reliability analysis of weighted–k–out–of–n systems, it is necessary to determine the order in which the weight of components is to be considered. The k–out–of–n :G(F) system consists of n components; each component has its own probability and positive integer weight such that the system is operational (failed) if and only if the total weight of some operational (failure) components is at least k. This paper designs a method to compute the reliability in $O(n \cdot k)$ computing time and in $O(n \cdot k)$ memory space. The proposed method expresses the system reliability in fewer product terms than those already published.

Keywords: Weighted–k–out–of–n system; Reliability equation; Order of components; Weight of components; Terms used in later step.

1. Introduction

The weighted–k–out–of–n :G(F) system consists of n components, each of which has its own probability and positive integer weight (total system weight = w), such that the system is operational (failed) if and only if the total weight of some operational (failure) components is at least k. The reliability of the weighted–k–out–of–n :G system is the component of the unreliability of a weighted–($w-k+1$)–out–of–n :F system. Without loss of generality, we discuss the weighted–k–out–of–n :G system only. The original k–out–of–n :G system is a special case of the weighted–k–out–of–n :G system wherein the weight of each component is 1.

One of the questions which arises when using recursive disjoint products algorithms for reliability of the weighted–k–out–of–n system is the order in which the weight of components should be considered [3]. The system was introduced by Wu and Chen in 1994 [1]. They proposed $O(n \cdot k)$ algorithm to compute the exact system reliability. However, their algorithm does not takes any account of the order of components. The number of product terms in their reliability equation is strongly influenced by the order of components.

Higashiyama has pointed out the advantages of an alternative order in the method based on the weight of components [2]. Three types of order are considered in [2]. One is the random order of components [1]. Second is to order the components so that the lower weight has younger component number, that is, in the order of the weight of components in the system, called best order or ascending order. For example, if the weight of component i is less than the weight of j, the component number i must be lower than the number j. Third is to order them that the weighty ones are first, called worst order or descending order. The best order method reduces the computing cost and data processing effort required to generate an optimal factored formula, which contains no identical terms [2].

The method [2] dramatically reduced the computing cost and data processing effort, however a lot of term unused in later steps are automatically derived in the method. This paper gives an efficient algorithm to generate the product terms only to be used in later steps.

Section 2 describes the notation & assumptions. Section 3 shows an $O(n \cdot k)$ algorithm by Wu–Chen for the reliability of the weighted–k–out–of–n :G system. Section 4 shows a revised algorithm by Higashiyama to generate a factored reliability formula. Section 5 proposes a new algorithm to reduce the number of computing steps.

2. Model

Notation

n	number of components in a system.
k	minimal total weight of all operational (failure)

	components which makes the system operational (failure).
w_i	weight of component i.
p_i	operational probability of component i.
q_i	$\triangleq 1.0 - p_i$, failure probability of component i.
R,B,W	[random, best, worst] case in which the components of the system are ordered [randomly, the lower weight one has younger number, the higher weight one has younger number].
$R_\Omega(i,j)$	reliability formula of the weighted–j–out–of–i: G_Ω for $\Omega = R,B,W$ case.
$R_\Omega^N(i,j)$	reliability formula of $R_\Omega(i,j)$ for $\Omega = R,B,W$ case in new method which generates the product terms only to be used in later steps.
$M_\Omega(i,j)$	binary random value indicating the state of $R_\Omega^N(i,j)$ for $\Omega = R,B,W$ case in the each step.

Assumptions

A. Each component and the system is either operational or failed.

B. There is no repair.

C. For given n, all components are mutually statistically independent.

D. Sensing and switching of failure components out of the system is perfect.

E. Each component has its own positive integer weight and its own probability.

F. The system is operational if and only if the total weight of operational components is at least k.

3. Wu–Chen (random case) [1]

Wu and Chen [1] have presented an $O(n \cdot k)$ algorithm to evaluate the reliability of the weighted–k–out–of–n: G_R system.

To derive $R_R(i,j)$, the algorithm needs to construct the table with $R_R(i,j)$, for $i = 0,1,2,\dots,n$, and $j = 0,1,2,\dots,k$. Initially,

$$R_R(i,0) = 1.0, \quad \text{for } i = 0,1,2,\dots,n; \tag{1}$$
$$R_R(0,j) = 0.0, \quad \text{for } j = 1,2,\dots,k. \tag{2}$$

Furthermore, if $j < 0$, it is obvious that for any i:

$$R_R(i,j) = 1.0 \tag{3}$$

For $i = 1,2,\dots,n,$ and $j = 1,2,\dots,k,$ their algorithm generates each $R_R(i,j)$,

$$R_R(i,j) = \begin{cases} p_i \cdot R(i-1, j-w_i) + q_i \cdot R(i-1,j), \\ \qquad\qquad\qquad \text{if } j - w_i \ge 0; \\ p_i + q_i \cdot R(i-1,j), \ \text{otherwise.} \end{cases} \tag{4}$$

The following details the algorithm for computing $R(n,k)$.

By equation (4), if $n < k$ then $R(i,j) = 0.0.$ Otherwise by equation (1), and equation (2), the algorithm constructs row 1 and column 1 in the $R_R(i,j)$ table. Then, by equation (4) the algorithm constructs row 2, row 3, $\dots$, row $n+1$ in that order; $R_R(n,k)$ is eventually derived. Because the size of the $R_R(i,j)$ table is $(n+1)\cdot(k+1)$, the size of the sequential algorithm needs $O(n \cdot k)$ running time.

This method has a disadvantage in that the number of terms depends on the order of components. Hereafter it is referred to as random order method.

Consider a weighted–5–out–of–3: G_R system with weights; $w_1 = 2$, $w_2 = 6$, and $w_3 = 4$.

By equation (1), get column #1 wherein,

$$R_R(0,0) = R_R(1,0) = R_R(2,0) = R_R(3,0) = 1.0 \tag{5}$$

and by equation (2), get row #1 wherein,

$$R_R(0,1) = R_R(0,2) = R_R(0,3) = R_R(0,4) = R_R(0,5) = 0.0 \tag{6}$$

Therefore, by equation (4) rows #2, #3, and #4 are derived as follows:

Row #2;

$$R_R(1,1) = p_1 \cdot R_R(0,-1) + q_1 \cdot R_R(0,1) = p_1 \tag{7}$$
$$R_R(1,2) = p_1 \cdot R_R(0,0) + q_1 \cdot R_R(0,2) = p_1 \tag{8}$$
$$R_R(1,3) = p_1 \cdot R_R(0,1) + q_1 \cdot R_R(0,3) = 0.0 \tag{9}$$
$$R_R(1,4) = p_1 \cdot R_R(0,2) + q_1 \cdot R_R(0,4) = 0.0 \tag{10}$$
$$R_R(1,5) = p_1 \cdot R_R(0,3) + q_1 \cdot R_R(0,5) = 0.0 \tag{11}$$

Row #3;

$$R_R(2,1) = p_2 \cdot R_R(1,-5) + q_2 \cdot R_R(1,1) = p_2 + q_2 p_1 \tag{12}$$
$$R_R(2,2) = p_2 \cdot R_R(1,-4) + q_2 \cdot R_R(1,2) = p_2 + q_2 p_1 \tag{13}$$
$$R_R(2,3) = p_2 \cdot R_R(1,-3) + q_2 \cdot R_R(1,3) = p_2 \tag{14}$$
$$R_R(2,4) = p_2 \cdot R_R(1,-2) + q_2 \cdot R_R(1,4) = p_2 \tag{15}$$
$$R_R(2,5) = p_2 \cdot R_R(1,-1) + q_2 \cdot R_R(1,5) = p_2 \tag{16}$$

Row #4;

$$R_R(3,1) = p_3 \cdot R_R(2,-3) + q_3 \cdot R_R(2,1)$$
$$= p_3 + q_3 \cdot (p_2 + q_2 p_1) = p_3 + q_3 p_2 + q_3 q_2 p_1 \tag{17}$$
$$R_R(3,2) = p_3 \cdot R_R(2,-2) + q_3 \cdot R_R(2,2)$$
$$= p_3 + q_3 \cdot (p_2 + q_2 p_1) = p_3 + q_3 p_2 + q_3 q_2 p_1 \tag{18}$$
$$R_R(3,3) = p_3 \cdot R_R(2,-1) + q_3 \cdot R_R(2,3) = p_3 + q_3 p_2 \tag{19}$$
$$R_R(3,4) = p_3 \cdot R_R(2,0) + q_3 \cdot R_R(2,4) = p_3 + q_3 p_2 \tag{20}$$
$$R_R(3,5) = p_3 \cdot R_R(2,1) + q_3 \cdot R_R(2,5)$$
$$= p_3 \cdot (p_2 + q_2 p_1) + q_3 p_2 = p_3 p_2 + p_3 q_2 p_1 + q_3 p_2 \tag{21}$$

4. Higashiyama [2]

4.1 Best case

This section presents the best order of components so that the lower weight one has younger component number. The procedure can be processed after reordering of components. Hereafter it is referred to as best order method.

Therefore, consider the reliability formula for the reordered weighted–5–out–of–3: G_B system with weights; $w_1 = 2$,

$w_2 = 4$, $w_3 = 6$.

By equation (1), get column #1 wherein,

$$R_B(0,0) = R_B(1,0) = R_B(2,0) = R_B(3,0) = 1.0 \qquad (22)$$

and by equation (2), get row #1 wherein,

$$R_B(0,1) = R_B(0,2) = R_B(0,3) = R_B(0,4) = R_B(0,5) = 0.0 \qquad (23)$$

Therefore, by equation (4) rows #2, #3, and #4 are derived as follows:

Row #2;

$$R_B(1,1) = p_1 \cdot R_B(0,-1) + q_1 \cdot R_B(0,1) = p_1 \qquad (24)$$
$$R_B(1,2) = p_1 \cdot R_B(0,0) + q_1 \cdot R_B(0,2) = p_1 \qquad (25)$$
$$R_B(1,3) = p_1 \cdot R_B(0,1) + q_1 \cdot R_B(0,3) = 0.0 \qquad (26)$$
$$R_B(1,4) = p_1 \cdot R_B(0,2) + q_1 \cdot R_B(0,4) = 0.0 \qquad (27)$$
$$R_B(1,5) = p_1 \cdot R_B(0,3) + q_1 \cdot R_B(0,5) = 0.0 \qquad (28)$$

Row #3;

$$R_B(2,1) = p_2 \cdot R_B(1,-3) + q_2 \cdot R_B(1,1) = p_2 + q_2 p_1 \qquad (29)$$
$$R_B(2,2) = p_2 \cdot R_B(1,-2) + q_2 \cdot R_B(1,2) = p_2 + q_2 p_1 \qquad (30)$$
$$R_B(2,3) = p_2 \cdot R_B(1,-1) + q_2 \cdot R_B(1,3) = p_2 \qquad (31)$$
$$R_B(2,4) = p_2 \cdot R_B(1,0) + q_2 \cdot R_B(1,4) = p_2 \qquad (32)$$
$$R_B(2,5) = p_2 \cdot R_B(1,1) + q_2 \cdot R_B(1,5) = p_2 p_1 \qquad (33)$$

Row #4;

$$R_B(3,5) = p_3 \cdot R_B(2,-1) + q_3 \cdot R_B(2,5) = p_3 + q_3 p_2 p_1 \qquad (34)$$

The final result $R_B(3,5)$ is only generated from reliabilities $R_B(2,-1)$ and $R_B(2,5)$, so it is not necessary to calculate $R_B(3,1)$, $R_B(3,2)$, … , $R_B(3,4)$.

4.2 Worst case

This section presents the worst order of components so that the higher weight one has younger component number. The procedure can be processed similar to best case after reordering of components. Hereafter it is referred to as worst order method.

Therefore, consider the reliability formula for the reordered weighted–5–out–of–3: G_W system with weights; $w_1 = 6$, $w_2 = 4$, $w_3 = 2$.

By equation (1), get column #1 wherein,

$$R_W(0,0) = R_W(1,0) = R_W(2,0) = R_W(3,0) = 1.0 \qquad (35)$$

and by equation (2), get row #1 wherein,

$$R_W(0,1) = R_W(0,2) = R_W(0,3) = R_W(0,4) = R_W(0,5) = 0.0 \qquad (36)$$

Therefore, by equation (4) rows #2, #3, and #4 are derived as follows:

Row #2;

$$R_W(1,1) = p_1 \cdot R_W(0,-5) + q_1 \cdot R_W(0,1) = p_1 \qquad (37)$$
$$R_W(1,2) = p_1 \cdot R_W(0,-4) + q_1 \cdot R_W(0,2) = p_1 \qquad (38)$$
$$R_W(1,3) = p_1 \cdot R_W(0,-3) + q_1 \cdot R_W(0,3) = p_1 \qquad (39)$$
$$R_W(1,4) = p_1 \cdot R_W(0,-2) + q_1 \cdot R_W(0,4) = p_1 \qquad (40)$$
$$R_W(1,5) = p_1 \cdot R_W(0,-1) + q_1 \cdot R_W(0,5) = p_1 \qquad (41)$$

Row #3;

$$R_W(2,1) = p_2 \cdot R_W(1,-3) + q_2 \cdot R_W(1,1) = p_1 + q_2 p_1 \qquad (42)$$
$$R_W(2,2) = p_2 \cdot R_W(1,-2) + q_2 \cdot R_W(1,2) = p_2 + q_2 p_1 \qquad (43)$$
$$R_W(2,3) = p_2 \cdot R_W(1,-1) + q_2 \cdot R_W(1,3) = p_2 + q_2 p_1 \qquad (44)$$
$$R_W(2,4) = p_2 \cdot R_W(1,0) + q_2 \cdot R_W(1,4) = p_2 + q_2 p_1 \qquad (45)$$
$$R_W(2,5) = p_2 \cdot R_W(1,1) + q_2 \cdot R_W(1,5) = p_2 p_1 + q_2 p_1 \qquad (46)$$

Row #4;

$$\begin{aligned} R_W(3,5) &= p_3 \cdot R_W(2,3) + q_3 \cdot R_W(2,5) \\ &= p_3 \cdot (p_2 + q_2 p_1) + q_3 \cdot (p_2 p_1 + q_2 p_1) \\ &= p_3 p_2 + p_3 q_2 p_1 + q_3 p_2 p_1 + q_3 q_2 p_1 \qquad (47) \end{aligned}$$

In the same manner to best case, the final result $R_W(3,5)$ is only generated from reliabilities $R_W(2,3)$ and $R_W(2,5)$, so it is not necessary to calculate $R_W(3,1)$, $R_W(3,2)$, … , $R_W(3,4)$.

4.3. Comparisons between three results

A. Using the component numbers in the weighted–5–out–of–3: G_B system, $R_R(3,5)$ (interchange component numbers 2 and 3) and $R_W(3,5)$ (interchange component numbers 1 and 3) can be rewritten as, respectively;

$$\begin{aligned} R_R(3,5) &= p_3 p_2 + q_3 p_2 p_1 + p_3 q_2 \\ &= p_3 \cdot (p_2 + q_2) + q_3 p_2 p_1 \\ &= p_3 + q_3 p_2 p_1 = R_B(3,5) \qquad (48) \end{aligned}$$

$$\begin{aligned} R_W(3,5) &= p_2 p_1 + p_3 q_2 p_1 + p_3 p_2 q_1 + p_3 q_2 q_1 \\ &= q_3 p_2 p_1 + p_3 p_2 p_1 + p_3 q_2 p_1 + p_3 p_2 q_1 + p_3 q_2 q_1 \\ &= p_3 \cdot \{(p_2 + q_2) \cdot p_1 + (p_2 + q_2) \cdot q_1\} + q_3 p_2 p_1 \\ &= p_3 + q_3 p_2 p_1 = R_B(3,5) \qquad (49) \end{aligned}$$

B. Best order method generates only 2 product terms and 4 variables, and requires 1 addition ($+$ –operator) and 2 multiplications ($\times$ –operator).

C. Random order method generates 3 product terms and 7 variables, and requires 2 additions and 4 multiplications.

D. Worst order method generates 4 product terms and 11 variables, and requires 3 additions and 7 multiplications.

5. Proposed method

5.1 Algorithm

The algorithm, **Generate disjoint product terms** in Fig.1, is based on the proper definition of the system structure function, which is given in *Notation* of Section 2. The format of the algorithm makes it easy to implement in a high–level programming language like Fortran, Pascal, or C.

5.2 Examples

Consider the weighted–5–out–of–3: G system with weights; $w_1 = 2$, $w_2 = 6$, $w_3 = 4$. For each case (R, B, W), the algorithm, **Generate disjoint product terms** in Fig.1, generates the disjoint product terms below for each case about the example system.

```
Procedure Generate disjoint product terms
  input: $n, k, w_1 \sim w_n, p_1 \sim p_n$;
  common: $n, k, w_1 \sim w_n, p_1 \sim p_n, M, R; q_i = 1.0 - p_i$;
  initial clear: $M[1 \le i \le n, 1 \le j \le k] := 0$;
    procedure Indicating matrix
      $M[n,k] := M[n-1,k] := 1$;
      if $k - w_n > 0$ then $M[n-1, k-w_n] := 1$; end if;
      for $i := n-1$ step $-1$ until $2$ do
        for $j := 1$ until $k$ do
          if $M[i,j] = 1$ then $M[i-1,j] := 1$;
            if $j - w_i > 0$ then $M[i-1, j-w_i] := 1$;
              end if; end if; end for; end for;
      Disjoint terms;
    end Indicating matrix
    procedure Disjoint terms
      initial clear: $R[0 \le i \le n, j \le 0] := 1.0$;
                     $R[0, 1 \le j \le k] := 0.0$;
      for $i := 1$ until $n$ do
        for $j := 1$ until $k$ do
          if $M[i,j] = 1$ then
            $R[i,j] := p_i \cdot R[i-1, j-w_i] + q_i \cdot R[i-1, j]$;
            end if; end for; end for;
    end Disjoint terms
  Indicating matrix;
end Generate disjoint product terms
```

Fig.1 Algorithm **Generate disjoint product terms**

5.2.1 Random case

After executing of **procedure Indicating matrix**, indicating matrix is;

$$M_R = \begin{bmatrix} 10001 \\ 10001 \\ 00001 \end{bmatrix}$$

By virtue of M_R, **procedure Disjoint terms** generates the reliability formulas as follows;

$$R_R^N(1,1) = p_1 \cdot R_R^N(0,-1) + q_1 \cdot R_R^N(0,1) = p_1 \qquad (7)'$$
$$R_R^N(1,5) = p_1 \cdot R_R^N(0,3) + q_1 \cdot R_R^N(0,5) = 0.0 \qquad (11)'$$
$$R_R^N(2,1) = p_2 \cdot R_R^N(1,-5) + q_2 \cdot R_R^N(1,1) = p_2 + q_2 p_1 \qquad (12)'$$
$$R_R^N(2,5) = p_2 \cdot R_R^N(1,-1) + q_2 \cdot R_R^N(1,5) = p_2 \qquad (16)'$$

Finally the algorithm derives the final result as follows;

$$\begin{aligned} R_R^N(3,5) &= p_3 \cdot R_R^N(2,1) + q_3 \cdot R_R^N(2,5) \\ &= p_3 \cdot (p_2 + q_2 p_1) + q_3 p_2 = p_3 p_2 + p_3 q_2 p_1 + q_3 p_2 \end{aligned} \qquad (21)'$$

5.2.2 Best case

After executing of **procedure Indicating matrix**, indicating matrix is;

$$M_B = \begin{bmatrix} 10001 \\ 00001 \\ 00001 \end{bmatrix}$$

By virtue of M_B, **procedure Disjoint terms** derives the reliability formulas as follows;

$$R_B^N(1,1) = p_1 \cdot R_B^N(0,-1) + q_1 \cdot R_B^N(0,1) = p_1 \qquad (24)'$$
$$R_B^N(1,5) = p_1 \cdot R_B^N(0,3) + q_1 \cdot R_B^N(0,5) = 0.0 \qquad (28)'$$
$$R_B^N(2,5) = p_2 \cdot R_B^N(1,1) + q_2 \cdot R_B^N(1,5) = p_2 p_1 \qquad (33)'$$
$$R_B^N(3,5) = p_3 \cdot R_B^N(2,-1) + q_3 \cdot R_B^N(2,5) = p_3 + q_3 p_2 p_1 \qquad (34)'$$

5.2.3 Worst case

The M_W and R_W^N are derived as follows;

$$M_W = \begin{bmatrix} 10101 \\ 00101 \\ 00001 \end{bmatrix}$$

$$R_W^N(1,1) = p_1 \cdot R_W(0,-5) + q_1 \cdot R_W(0,1) = p_1 \qquad (37)'$$
$$R_W^N(1,3) = p_1 \cdot R_W^N(0,-3) + q_1 \cdot R_W^N(0,3) = p_1 \qquad (39)'$$
$$R_W^N(1,5) = p_1 \cdot R_W^N(0,-1) + q_1 \cdot R_W^N(0,5) = p_1 \qquad (41)'$$
$$R_W^N(2,3) = p_2 \cdot R_W^N(1,-1) + q_2 \cdot R_W^N(1,3) = p_2 + q_2 p_1 \qquad (44)'$$
$$R_W^N(2,5) = p_2 \cdot R_W^N(1,1) + q_2 \cdot R_W^N(1,5) = p_2 p_1 + q_2 p_1 \qquad (46)'$$
$$\begin{aligned} R_W^N(3,5) &= p_3 \cdot R_W^N(2,3) + q_3 \cdot R_W^N(2,5) \\ &= p_3 \cdot (p_2 + q_2 p_1) + q_3 \cdot (p_2 p_1 + q_2 p_1) \\ &= p_3 p_2 + p_3 q_2 p_1 + q_3 p_2 p_1 + q_3 q_2 p_1 \end{aligned} \qquad (47)'$$

5.3 Comparisons

The proposed algorithm can generate three types of the final reliability function above, equation $R_R^N(3,5)$ in (21)', equation $R_B^N(3,5)$ in (34)', or equation $R_W^N(3,5)$ in (47)', for each case.

A. For the random case, the proposed algorithm needs 5 reliability formulas to get the final reliability function and 6 reliability formulas are omitted.
B. For the best case, the proposed algorithm needs 4 reliability formulas to get the final reliability function and 7 reliability formulas are omitted.
C. For the worst case, the proposed algorithm needs 6 reliability formulas to get the final reliability function and 5 reliability formulas are omitted.

References

[1] J.S. Wu and R.J. Chen, “An algorithm for computing the reliability of weighted–k–out–of–n systems”, *IEEE Transactions on Reliability*, Vol.43(2), pp327–329 (1990).

[2] Y. Higashiyama, “A factored reliability formula for weighted–k–out–of–n system”, *Asia–Pacific Journal of Operational Research*, Vol.18, pp61–66(2001).

[3] J.A. Buzacott, “The ordering of terms in cut– based recursive disjoint products”, *IEEE Transactions on Reliability*, Vol.R–32(5), pp472– 474(1983).