

## A Path-Set Based Approach for Two-Terminal Reliability Computation of Interconnection Networks

<sup>1</sup>Pradyumna Kumar Tripathy, <sup>2</sup>Ranjan Kumar Dash,  
<sup>3</sup>Rabindra Kumar Dalei and <sup>4</sup>Chitta Ranjan Tripathy

<sup>1</sup>Department of Computer Science and Engineering, Silicon Institute of Technology,

<sup>2</sup>Department of Information and Technology, College of Engineering and Technology,

<sup>3</sup>Department of Master of Computer Application, Silicon Institute of Technology,  
Bhubaneswar, Odisha, India

<sup>4</sup>Department of Computer Science and Engineering,  
Veer Surendra Sai University of Technology, Burla, Odisha, India

---

**Abstract:** Design of a reliable and cost effective interconnection network is highly desirable in parallel computing environments. The reliability in particular plays an important role in design of such interconnection networks. Among different reliability measures, the terminal reliability or two-terminal reliability is an important reliability measure as it ensures a reliable path between a pair of source and destination nodes in a interconnection network under different failures. However, the exact estimation of two-terminal reliability is a NP-hard problem. This study proposes a new and efficient minimal path-set based approach for exact estimation of two-terminal reliability. The proposed approach not only generates all non-redundant minimal path sets but also efficiently computes the reliability of the interconnection network with greater accuracy. The proposed method is well illustrated by taking a simple example interconnection network. The simulated results further ensures its applicability to different kinds of interconnection networks viz. directed, undirected, homogenous, non-homogeneous, regular and general networks.

**Key words:** Interconnection networks, reliability, minimal path-set, probabilistic graph, parallel computing system, efficiently

---

### INTRODUCTION

There is always a need for some kind of communication highway or interconnection network in parallel computing system in order to solve a problem where the processors needs to be connected in some pattern. Selection of right interconnection network is important for efficiency reasons as performance in multiprocessor systems is highly dependent on communication processes between processors and memory, I/O devices and other processors. The factors those affects the design of interconnection networks are reliability, cost, flow, power consumption, lead time, etc. Among which reliability is the most important factor to be considered while selection of efficient interconnection network. Because it is the ability of a system to consistently perform its intended or required function without degradation or failure in a predefined time (Colbourn and Colbourn, 1987). The

different reliability measures considered in the design of interconnection networks are network reliability two-terminal reliability and k-terminal reliability. The two-terminal reliability is considered to be a one the most important reliability measure in interconnection network as it is the probability that a communication exists between a specified pair of nodes in network.

A recursive truncation algorithm for estimating all-terminal network reliability is proposed by Sharafat and Marouzi. (2009). They used a recursive approach by collapsing the nodes of the network in every iteration by gradually reducing the size of the network. In recent research, Rodionov *et al.* (2012) also proposed a new network reliability estimation algorithm where they considered improvements in the efficiency of cumulative updating of all-terminal network reliability. By Hardy *et al.* (2007), used binary decision diagrams for reliability estimation of interconnection networks. These methods only give emphasis on network reliability. There are

various approaches reported in literature for two terminal reliability estimation of interconnection networks viz. cut-set based approach (Chen and Yuang, 1996; Tripathy *et al.*, 2014; Yeh, 2008), path-set based approach (Balan and Traldi, 2003; Nahman, 1994; Yeh, 2007, 2009; Chen, 2011) binary decision diagrams (Kuo *et al.*, 1999; Zhang *et al.*, 2003), sum of disjoint product (Balan and Traldi, 2003), enumerative approach (Yeh, 2009).

Chen and Yuang (1996) proposed a cut-set based approach for two-terminal reliability computation of interconnection networks. A similar approach is also found in literature (Kuo *et al.*, 1999) for terminal pair reliability estimation by using edge expansion diagrams using OBDD. They considered all possible edge expansion of the network and use the binary decision diagrams technique to compute the terminal pair reliability of the interconnection networks. The multi-state reliability also has drawn the much attention of many researchers and several approaches are reported in the literature (Jane and Lai, 2008, 2010; Zhang *et al.*, 2003). A practical algorithm for computing multi-state two-terminal reliability is proposed by Jane and Lai (2008). Zhang *et al.* (2003) proposed a BDD-based algorithm for analysis of multi state systems. They considered the multi state systems with multi state components. The exhaustive search algorithm for finding such ordering is not practical for most real world applications. Jane and Laith (2010) proposed an algorithm for computing multi-state two-terminal reliability. They computed the reliability through critical arc states that interrupt demand. In recent, Tripathy *et al.* (2014), proposed a self generating disjoint minimal cut-set based approach for evaluating different reliability measures of interconnection networks. By Tripathy *et al.* (2013), a Genetic algorithm based approach is used for reliability optimization of interconnection networks where a new approach for network reliability estimation is discussed.

There are some minimal path-set based approaches found in the literature for two-terminal reliability estimation of interconnection networks (Balan and Traldi, 2003; Nahman, 1994; Yeh, 2007, 2009; Chen, 2011). Balan and Traldi (2003) applied the sum of disjoint products technique on pre-processed minimal paths to obtain the success probability of the network. An enumerative based approach is proposed by Nahman (1994) to generate all minimal paths of modified networks. Yeh (2009) proposed a new method for finding all minimal paths in a network by using simple universal generating function. A heuristic based algorithm for generating all minimal paths is proposed by Yeh (2007). In recent, Chen (2011) proposed a method for generating all minimal paths

of a network considering failure of nodes. However in this technique, the size of the network increases with the no. of links in the network and the networks need to have directed edges.

Lin (2002) presented a minimal cut-set based approach for reliability estimation of networks. They considered the stochastic-flow network with failures at nodes and arcs. Similar approach is found by Yan and Qian (2007) for improving efficiency of solving the minimal cut problem in stochastic-flow networks. There is general assumption in the literature that there are two states of a system. However in a practical case there are more than just the binary state. Hence, the multi state two-terminal reliability problems come into picture. An improved algorithm for searching all multi-state minimal cuts (Yeh, 2008) and to search for all minimal-cuts of a limited-flow network (Salehi-Fathabadi and Forghani-elahabadi, 2009) is found in the literature. Dash *et al.* (2012) proposed a efficient model for reliability optimization of interconnection networks where they considered the failure of both node and edge of the network. By Tripathy *et al.* (2015), a layout optimization of interconnection networks is discussed. They used dynamic programming approach for solving the said problem. Lin and Chang (2011) proposed a method for maintenance reliability estimation for a cloud computing network considering the nodes failure. Kuo *et al.* (2007) proposed an efficient and exact reliability evaluation for networks considering failure of vertices.

There is lots of research carried out for computation of different reliability measures of different kinds of networks. However, the existing approaches have limitations like the symbolic expression/augmentation based algorithms has inefficient search in both computation and storage time. Additional overhead of finding the duplicate elements occurred in direct search based algorithms. Some of the techniques exhibits exponential rise in the number of terms even for moderately sized networks. Many of the algorithm are only designed to suit for a particular kind of network. Therefore, it motivates us to design an efficient algorithm which could able to correctly compute the two-terminal reliability without generating intermediate duplicate paths.

## MATERIALS AND METHODS

### Proposed algorithm

**Problem definition:** The problem is for exact computation of two-terminal reliability of the given interconnection network through minimal path-set based approach. The interconnected network can be viewed as a probabilistic

graph where the nodes represent the vertices and the edges represent the connection links between them. Here, the probabilistic graph is represented in form a incidence matrix. The approach is to generate all possible minimal paths from a given source *s* to a destination *t* and then use this minimal paths to generate the reliability expression and to compute the Two-Terminal reliability of the interconnection network.

**Definitions**

**Minimal path:** A minimal path is the path that connects the source *s* to the sink *t* as long as:

- It contains no cycles
- Removal of any of the edges from the path means that there is no longer a connection between *s* and *t*

**Two-terminal reliability:** The probability that a communication path exists between a specified pair of nodes in network.

**Proposed algorithm:**

**Input:** *G* in the form of incidence matrix, source node *s* and terminal node *t*

**Output:** Computed reliability

Reliability-evaluation-path-set-method (*G*, *s*, *t*)

```

1. temp-arr ← G
2. for i = 1-L
3.   begin
4.     for j = 1-L
5.       begin
6.         Link [i] [j] ← destination of G (i, j)
7.       end
8.     end
9.   calc-rel (s)
10.  begin
11.    Push (arr, s) //push the upcoming node into the stack
12.    for (i = all edges of the directed matrix)
13.      begin
14.        if (link [s] [i] > 0)
15.          if ((top+1 < N) || inner-cycle()) //To check
16.            if link is present between node s to link i
17.              if (link [s] [i] = t) //if we can reach
18.                calculate() //Calculating the expression
19.                of generated minimal path
20.              end if
21.            end if
22.          else
23.            calc-rel (link [s] [i]) //Recursively calling
24.            the function
25.          endif
26.        else continue,
27.      end
28.    Pop (arr) //popping the top element
29.    //calculate reliability using
30.    the function
31.  end

```

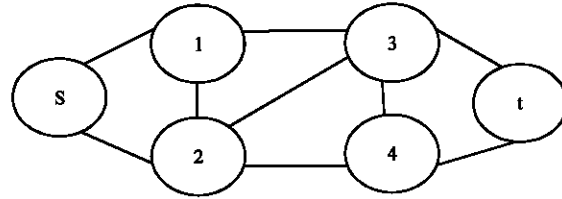


Fig. 1: Example interconnected network with 6 nodes and 9 links

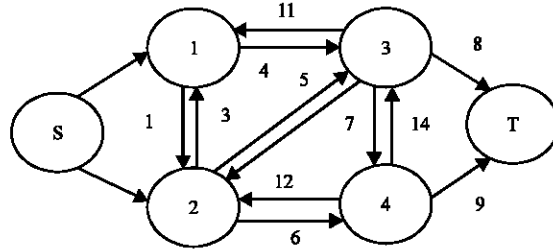


Fig. 2: Directed form of graph

**Illustration**

**Input:** Let us, assume an Interconnected Network (ICN) with 6 nodes and 9 links. The equivalent probabilistic graph *G* is given in Fig. 1. The network in Fig. 1 is an undirected network. The adjacency matrix for the given network is as follows:

/	s	1	2	3	4	t
s	0	1	1	0	0	0
1	1	0	1	1	0	0
2	1	1	0	1	1	0
3	0	1	1	0	1	1
4	0	0	1	1	0	1
t	0	0	0	1	1	0

**Process:** The undirected graph is converted to its directed form. The directed form of the example network with six node and 9 links (Fig. 1) is presented in Fig. 2. Since, the proposed algorithm requires path from *s*-*t*, the outgoing links from *s* and incoming links to *t* are considered unidirectional in this figure while other edges are considered as bidirectional.

From the above directed graph a matrix is obtained in which from *i*th (row) node using *j*th (column) link the destination node can be reached (Fig. 2).

/	1	2	3	4	5	6	7	8	9	10	11	12	13	14
S	1	2	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	3	0	0	0	0	0	2	0	0	0	0
2	0	0	1	0	3	4	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	4	t	0	0	1	2	0	0
4	0	0	0	0	0	0	0	0	t	0	0	0	2	3

**Calc-rel (s):** Calc-rel function is implemented on the matrix by passing the source node as the parameter initially.

**Push (arr, s):** Upcoming node is pushed into the stack. Stack status: s for  $(i = 0-i = 14)/14$  is the number of edges in the converted directed form of the input network. The destination node for each upcoming node is checked by associating the link attached with it. If  $(link [s] [i] > 0)$ : Node 's' is associated with two links, i.e., 1 and 2. Link [s] (Colbourn and Colbourn, 1987) and link [s] (Sharafat and Ma'rouzi, 2009) will have values more than 0. We will go on with  $i = 1$  initially. If  $((top)+1 < N)$ : The value of  $top = 1$ . The expression is true. If  $(link [s] [i] = t)$ : As value of  $i = 1$ , the value of link [s] (Colbourn and Colbourn, 1987) is checked and it is not equal to the terminal node. Therefore, the 'if' block is not executed. else  $calc-rel (link [s] [i])$ : The destination node is passed as an argument in the function and called recursively till the destination node and the terminal node are same. The value of link [s] (Colbourn and Colbourn, 1987) is 1 so therefore we call  $calc-rel (1)$ .

So, at this point of time only 's' i.e., the source node is placed in the stack. The function is called recursively and the value of s is now 1. Therefore,  $calc-rel (1)$ .

**Push (arr, s):** Upcoming node is pushed into the stack. Stack status: s, 1. For  $(i = 0-i = 14)$ : The destination node for each upcoming node and the link associated with it is checked. If  $(link [s] [i] > 0)$ : In the case for node 's = 1' two links are used: 4 and 10. So, link [s] (Hardy et al., 2007) and link [s] (Balan and Traldi, 2003) will have values more than 0. If  $((top)+1 < N)$ . As the value of  $top = 2$  therefore this expression is found to be true. If  $(link [s] [i] = s)$ : As value of  $i = 4$ , the value of link [s] (Hardy et al., 2007) is checked. But it is not equal to the destination node. Therefore, the if block is not executed else  $calc-rel (link [s] [i])$ : The destination node is passed through the function and called, recursively till the destination node and the terminal node are same. The value of link [s] (Hardy et al., 2007) is 3, so therefore, we call  $calc-rel (3)$ . So, stack status is source node s and node 1. The function is recursively called and the value of s is now 3. So,  $calc-rel (3)$  is evaluated. Push (arr, s): Upcoming node is pushed into the stack. Stack status is now s, 1, 3.

**For (i = 0-i = 14):** The destination node for each upcoming node and the link associated with it is checked. If  $(link [s] [i] > 0)$ : here for node 's = 3' four links are there: 7, 8, 11 and 12. So, link [s] (Jane and Laih, 2008), link [s] (Zhang et al., 2003), link [s] (Nahman, 1994) and link [s] (Yeh, 2009) will have values more than 0. The first value of 'i' that satisfies is 4 so we go on with  $i = 7$ .

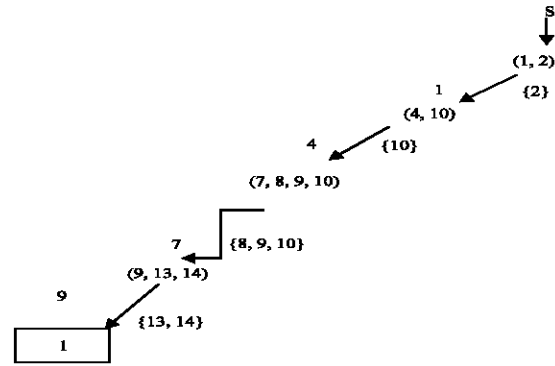


Fig. 3: Intermediate tree structure of path generation process

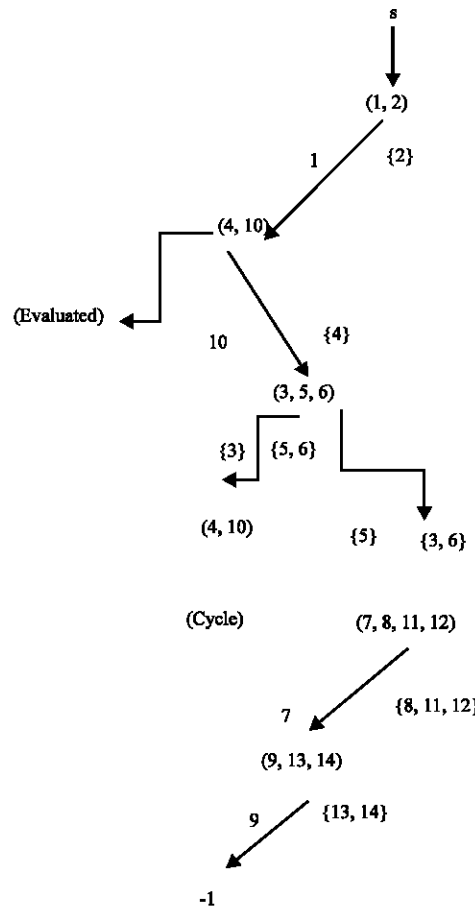


Fig. 4: The intermediate tree structure for computing all minimal paths of G

**If  $((top)+1 < N)$ :** As the value of  $top = 3$ , therefore, this expression is evaluated to true. If  $(link [s] [i] = t)$ . As value of  $i = 7$ , therefore the value of link [s] (Jane and Laih, 2008) is checked and it is not equal to the destination node. Else  $calc-rel (link [s] [i])$ : The destination node is passed in the function as parameter and called recursively till the

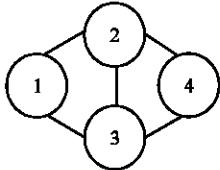
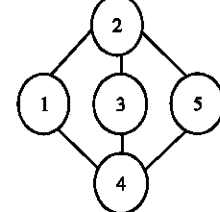
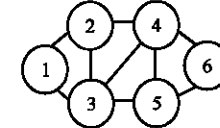
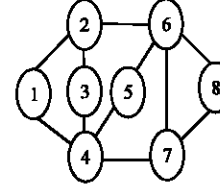
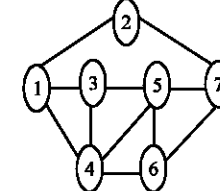
Table 1: Content of p-arr after all minimal path generation

Steps	Value of p-arr
0	4
1	3
2	5
3	5
4	4
5	4
6	5
7	5
8	4
9	4
10	3
11	3
12	4

Table 2: Content of q-arr after all minimal path generation

Steps	Value of q-arr
0	7
1	5
2	9
3	9
4	7
5	6
6	9
7	9
8	7
9	8
10	6
11	5
12	8

Table 3: Application of proposed algorithm on some sample general interconnection networks for two-terminal reliability evaluation

Networks	Minimal paths	Expression	Reliability
	{1, 3, 5}, {2, 4}, {2, 5}, {2, 3, 4}	$p3q3+p2q2+p3q3+p2q2$	0.981
	{1, 6}, {1, 9, 5, 8}, {1, 9, 7}, {3, 8}, {3, 10, 7}, {3, 10, 4, 6}, {2, 4, 6}, {2, 5, 8}, {2, 7}	$p2q3+p4q6+p3q5+p2q3+$ $p3q5+p4q6+p3q5+p3q5+$ $p2q4$	0.983
	{1, 4, 7, 9}, {1, 4, 8}, {1, 10, 5, 7, 9}, 1, 10, 5, 8}, {1, 10, 6, 9}, {1, 10, 6, 14, 8}, 2, 3, 4, 7, 9}, {2, 3, 4, 8}, {2, 5, 7, 9}, {2, 5, 8}, {2, 6, 9}, {2, 6, 14, 8}	$p4q7 + p3q5+p5q9+p4q7+$ $p4q6+p5q9+p5q9+p4q7$ $+p4q8+ p3q6+ p3q5+$ $p4q8 +p5q9$	0.978
	{1, 9}, {2, 4, 8, 11}, {2, 4, 10}, {2, 7, 11}, {2, 7, 15, 10}, {2, 12, 16, 11}, {2, 12, 16, 15, 10}, {3, 5, 4, 8, 6}, {3, 5, 4, 10}, {3, 5, 7, 11}, {3, 5, 7, 15, 10}	$p2q2 +p4q9+p3q6+p3q7+$ $p4q10+p4q7+p5q10+p5q10+$ $p4q7+ p4q8+ p5q10+$ $p3q6 +p5q10+p4q8$	0.987
	{3, 16, 11}, {3, 16, 13, 4, 10}, {3, 16, 15, 10} {1, 3, 10}, {1, 3, 9, 11}, {1, 3, 17, 16, 6, 11}, {1, 13, 14, 6, 11}, {1, 13, 14, 6, 8, 10}, {1, 13, 14, 7, 8, 10}, {1, 13, 14, 7, 8, 9, 11}, {2, 5, 4, 3, 9, 11}, {2, 5, 4, 3, 10}, {2, 6, 11}, {2, 6, 18, 10}, {2, 7, 8, 9, 11}, {2, 7, 8, 10}	$p3q5+p4q7+p6q10+$ $p5q7+p6q10+p6q10+p7q12+$ $p6q10+p5q9+p3q5+$ $p4q8+p5q+p4q$	0.985

destination node and the terminal node are same. The value of link [s] (Jane and Laih, 2008) is 4 so calc-rel (4) is called. Now, the stack contains source node, node 1 and

node 3. The function is called recursively and the value of s is now 4. So, calc-rel (4) is called. Push (arr, s): the upcoming node is pushed into the stack. Stack status is

now s, 1, 3, 4. For (i = 0-i = 14): the destination node for each upcoming node and the link associated with it is checked.

**If (link [s] [i]>0):** In the case for node 's = 4' three links are there: 9, 13 and 14. So, link [s] (Jane and Laih, 2010), link [s] (Yeh, 2007) and link [s] (Chen, 2011) will have values more than 0. The first value of 'T' satisfies with 9 so we go on with i = 9. If ((top)+1 < N): As the value of top = 4 the expression is evaluated to true.

**If (link [s] [i] = s):** The value of i = 9, now the value of link [s] (Jane and Laih, 2010) same as the terminal node. So, the if block is executed and the reliability expression is found out by raising the value of top to the power of 'p' and raising the value of (sum of all the links present in the stack the value of top) to the power of q. Now, 1 minimal path is found out and the expression for it is stored in p-arr and q-arr. The stack contains source node, node 1, 3 and 4 and Diagrammatically the generation process can be seen in form of a tree like structure presented in Fig. 3.

Similar process can be carried out for finding all 13 minimal paths of the graph s. The intermediate tree generated is for finding all minimal paths is shown in Fig. 4. At the same time the value for the unreliability expression is stored in p-arr and q-arr. The final content of p-arr and q-arr is given in Table 1 and 2, respectively. Finally, the p-arr and q-arr can be used to compute the reliability.

The two-terminal unreliability expression is computed as:  $UR = p^4 q^7 + p^3 q^5 + p^5 q^9 + p^5 q^9 + p^4 q^7 + p^4 q^6 + p^5 q^9 + p^5 q^9 + p^4 q^7 + p^4 q^8 + p^3 q^6 + p^3 q^5 + p^4 q^8$ . Considering the value of p as 0.9 and q as 0.1 and  $R = 1 - UR$ . The two-terminal reliability of the interconnection network ICN comes out to be becomes 0.978.

**RESULTS AND DISCUSSION**

The proposed algorithm is applied on some sample interconnection networks and simulated on MATLAB 2009 to show its correctness and efficiency. The result is presented in Table 3.

It can be clearly observed from Table 3 that the proposed algorithm efficiently generates all minimal path-sets of the input interconnection networks. It also forms the unreliability expression by considering the minimal path-sets with respect to the failure of the possible edges. The reliability value is then computed by the formula  $R = 1 - UR$ . It is also quite clear from the observation that the proposed approach never generates

duplicate path-sets which guarantees the correct evaluation of two-terminal reliability and its efficiency.

**CONCLUSION**

A new and efficient algorithm for exact evaluation of two-terminal reliability is proposed in this study. The proposed approach uses minimal path-sets for two-terminal reliability evaluation. It generates all non-redundant minimal path-sets and implicitly stores the reliability and unreliability value for each path-set for reliability computation. By so, it overcomes the additional overhead of disjointing process of minimal paths. The proposed approach is suitable for different kinds of interconnection networks viz. regular, general, directed, undirected, homogenous and non-homogeneous interconnection networks. The presented approach may be extended to evaluate two terminal reliabilities of stochastic flow networks as well as other such real time networks. Further, other reliability measures viz. k-terminal reliability and all terminal reliability can also be evaluated by modifying the proposed research in this study.

**Notations:**

- ICN = Input interconnection network
- G = Equivalent probabilistic Graph of ICN in incident matrix form
- N and L = Total number of Nodes and Links in G, respectively
- temp-arr = Matrix for storing directed edge form of the graph
- s = Source node
- t = Terminal node
- arr = Stack for storing the links/edges while finding the minimal paths
- link [I] [j] = Returns destination node by taking current node i and edge j associated with it
- p-arr = Matrix for storing the exponents of unreliability expression
- q-arr = Matrix for storing the exponents of unreliability expression
- inner-cycle() = Returns 1 if the node is present in the stack more than once otherwise 0
- calc-rel () = Calculate the reliability of the interconnected network from the p-arr and the q-arr
- calculate() = Calculating the expression of generated minimal path
- pop () = Deletion operation of stack
- UR and R = Unreliability and reliability of ICN, respectively

**REFERENCES**

- Balan, A.O. and L. Traldi, 2003. Preprocessing minpaths for sum of disjoint products. *IEEE. Trans. Reliab.*, 52: 289-295.
- Chen, S.G., 2011. Search for all minimal paths in a general directed flow network with unreliable nodes. *Intl. J. Reliab. Qual. Perform.*, 2: 63-70.
- Chen, Y.G. and M.C. Yuang, 1996. A cut-based method for terminal-pair reliability. *IEEE. Trans. Reliab.*, 45: 413-416.
- Colbourn, C.J. and C.J. Colbourn, 1987. *The Combinatorics of Network Reliability*. Vol. 200, Oxford University Press, New York, USA., Pages: 159.
- Dash, R.K., N.K. Badapanda, P.K. Tripathy and C.R. Tripathy, 2012. Reliability optimization of interconnection network node edge failure model. *Appl. Soft Comput.*, 12: 2322-2328.
- Hardy, G., C. Lucet and N. Limnios, 2007. K-terminal network reliability measures with binary decision diagrams. *IEEE. Trans. Reliab.*, 56: 506-515.
- Jane, C.C. and Y.W. Lai, 2008. A practical algorithm for computing multi-state two-terminal reliability. *IEEE. Transac. Reliab.*, 57: 295-302.
- Jane, C.C. and Y.W. Lai, 2010. Computing multi-state two-terminal reliability through critical arc states that interrupt demand. *IEEE. Trans. Reliab.*, 59: 338-345.
- Kuo, S.Y., F.M. Yeh and H.Y. Lin, 2007. Efficient and exact reliability evaluation for networks with imperfect vertices. *IEEE. Trans. Reliab.*, 56: 288-300.
- Kuo, S.Y., S.K. Lu and F.M. Yeh, 1999. Determining terminal-pair reliability based on edge expansion diagrams using OBDD. *IEEE. Trans. Reliab.*, 48: 234-246.
- Lin, Y.K. and P.C. Chang, 2011. Maintenance reliability estimation for a cloud computing network with nodes failure. *Expert Syst. Appl.*, 38: 14185-14189.
- Lin, Y.K., 2002. Using minimal cuts to evaluate the system reliability of a stochastic-flow network with failures at nodes and arcs. *Reliab. Eng. Syst. Saf.*, 75: 41-46.
- Nahman, J.M., 1994. Enumeration of mps of modified networks. *Micro Electron. Reliab.*, 34: 475-484.
- Rodionov, A., D. Migov and O. Rodionova, 2012. Improvements in the efficiency of cumulative updating of all-terminal network reliability. *IEEE. Trans. Reliab.*, 61: 460-465.
- Salehi-Fathabadi, H. and M. Forghani-elahabadi, 2009. A note on a simple approach to search for all-MCs of a limited-flow network. *Reliab. Eng. Syst. Saf.*, 94: 1878-1880.
- Sharafat, A.R. and O.R. Ma'rouzi, 2009. All-terminal network reliability using recursive truncation algorithm. *IEEE. Trans. Reliab.*, 58: 338-347.
- Tripathy, P.K., R.K. Dash and C.R. Tripathy, 2013. A new genetic algorithm based method for topological optimization of interconnection networks. *Intl. J. Comput. Appl.*, 63: 7-13.
- Tripathy, P.K., R.K. Dash and C.R. Tripathy, 2014. An efficient method based on self-generating disjoint minimal cut-sets for evaluating reliability measures of interconnection networks. *Intl. J. Performability Eng.*, 10: 303-312.
- Tripathy, P.K., R.K. Dash and C.R. Tripathy, 2015. A dynamic programming approach for layout optimization of interconnection networks. *Eng. Sci. Technol. Intl. J.*, 18: 374-384.
- Yan, Z. and M. Qian, 2007. Improving efficiency of solving MC problem in stochastic-flow network. *Reliab. Eng. Syst. Saf.*, 92: 30-39.
- Yeh, W.C., 2007. A simple heuristic algorithm for generating all minimal paths. *IEEE. Trans. Reliab.*, 56: 488-494.
- Yeh, W.C., 2008. A fast algorithm for searching all multi-state minimal cuts. *IEEE. Trans. Reliab.*, 57: 581-588.
- Yeh, W.C., 2009. A simple universal generating function method to search for all minimal paths in networks. *IEEE. Trans. Syst. Man Cybern. Part A. Syst. Humans*, 39: 1247-1254.
- Zang, X., D. Wang, H. Sun and K.S. Trivedi, 2003. A BDD-based algorithm for analysis of multistate systems with multistate components. *IEEE. Trans. Comput.*, 52: 1689-1691.