Network Reliability Evaluation with Changes in Layout

N. K. GOYAL

Reliability Engineering Centre, IIT Kharagpur, INDIA

(Received on September 3, 2008, revised on February 20, 2009)

Abstract: Network reliability evaluation techniques, e.g., path (cut) set techniques, factoring theorem based techniques etc., evaluate various network reliability measures based on different connectivity criterion of nodes, which is a NP hard problem. A little change in network layout requires repetition of the complete procedure. In this paper, a new approach based on path set technique is proposed. The proposed approach stores and process reliability expressions in terms of minimal path sets and disjoint sets using binary data structure. This paper proposes algorithms for modifying these sets with modifications in network layout. This paper also proposes a method to evaluate reliability defined on the basis of different node connectivity requirements.

Keywords: Network reliability, minimal path set, disjoint set, terminal pair reliability, k-terminal reliability, connectivity criterion based reliability.

1. Introduction

A primary measure of network performance is connectivity among nodes. Therefore, network reliability has been primarily addressed in literature based on connectivity among nodes. Three commonly used reliability measures based are: two terminal or terminal pair reliability (TPR), k-terminal reliability (KTR) and all terminal reliability (ATR) or global reliability [1].

Generally, a network user/designer is interested to know the system reliability not only for one node pair but for most of node pairs and if possible all node pairs of the network. He/she may also be interested not only for connectivity among a set of nodes but also for other user defined connectivity criterions. For example, probability that a set of nodes is connected with each other OR another set of nodes is connected with each other, as this may satisfy the network connectivity requirements. Besides, network layout may change frequently during design. In such cases, the general approach is re-evaluation of reliability indices, which requires a lot of time and efforts. This also includes repetition of efforts made during earlier evaluation. The time and efforts required for the evaluation and limited network reliability information provided by available techniques may be a reason for their limited use.

Available literature on reliability evaluation of communication networks can broadly be classified into two paradigms a) Path sets or Cut sets (POC) approaches paradigm and b) Non-Path sets or Cut sets (NPOC) approaches paradigm. A detailed discussion on these approaches is provided in [1-3]. NPOC paradigm approaches, e.g., SNEM [4], factoring theorem [5-9] and BDD approaches [10], have been applied for evaluating TPR and KTR.
However, all these approaches do not render a compact reliability expression. Moreover, these techniques use graph partitioning and transformation techniques, which may make it difficult to modify reliability expressions with network layout changes.

A new approach is proposed in this paper, which stores reliability expressions for node pairs of a network in a proposed data structure and modifies these expressions with changes in network layout. Using the modifications in place of re-evaluation is more efficient as significant amount of calculations required for generating sum of disjoint product (SDP) terms from minimal cut sets or path sets are not repeated. Reliability value for any user defined connectivity criterion can be obtained by applying proposed union (OR) and/or intersection (AND) operators on already evaluated node pair reliability expressions.

Notation

- \( N/L \): Number of nodes/links in the network.
- \( N_i/X_i \): \( i \)th node/link of any network, where \( 1 \leq i \leq N, \) or \( 1 \leq i \leq L. \)
- \( p_i/q_i \): Reliability/unreliability of \( i \)th link, where \( 1 \leq i \leq L. \)
- \( \text{RE}(i,j) \): Reliability expression for node pair \((N_i, N_j)\).
- \( +, * \): OR(Union) operator, AND(Intersection) operator, respectively.
- \( \sim, \oplus \): NOT, XOR operators.

Assumptions

1. The nodes of the network are perfectly reliable.
2. The network and its links have only two states (i) working or (ii) failed.
3. The link failures are statistically independent.
4. Links can communicate in both directions.

2. Proposed Modeling Approach

The reliability evaluation approach proposed in this paper utilizes a model data structure of node pair reliability expressions for networks. In this approach, the network under analysis is obtained by modifying an already evaluated network present in the model database. Following five basic modifications are considered in the proposed approach:

- **M1/M2:** Removing a set of links / nodes.
- **M3:** Adding a link to a new node.
- **M4:** Joining two network segments through a link.
- **M5:** Adding a link between two nodes of a network.

Pseudo-codes for all the above modifications are provided in [2].

The model data structure used for storing reliability expressions is presented next. It has been developed for achieving two objectives: minimizing memory requirements and maximizing execution efficiency using bit-strings and bitwise operators.

2.1 Data Structure

The data structure for reliability expressions of node pair \((N_i, N_j)\) and reliability expressions for all of node pairs of network shown in Figure 1 are given in Table 1. To express reliability expression in SDP form, following binary strings are implemented:

1. **Path Set (PS):** Minimal path set for a node pair \((i, j)\) is denoted as \( \text{RE}(i,j).\text{PS} \). Set of links and nodes in \( k \)th simple path are denoted by \( \text{RE}(i,j).\text{PS}(k).\text{PN} \) and \( \text{RE}(i,j).\text{PS}(k).\text{PL} \) binary strings, respectively.
2. **Disjoint Set (DS):** Sum of disjoint product (SDP) approaches make a simple path disjoint with preceding paths in the set. This disjointing procedure, suggested by Aggarwal et al. [11] and later modified by Abraham [12], generates terms, which
are added together and multiplied with the path. These multiplying terms, referred as disjoint terms, consists of link reliability and unreliability terms. The reliability and unreliability terms for \(i^{th}\) disjoint term are stored in two parts as \(RE(i,j).PS(k).DSR\) and \(RE(i,j).PS(k).DSQ\) respectively.

![Figure 1: A Typical Network](image1)

![Figure 2: After Deleting link \(X_1\) from Figure 1](image2)

![Figure 3: After Deleting node \(N_4\) from Figure 1](image3)

Table 1: Data Structure and Reliability Expressions for Network shown in Figure 1

<table>
<thead>
<tr>
<th>((N_i,N_j))</th>
<th>Path Set (PS)</th>
<th>Disjoint Set (DS)</th>
<th>(RE(i,j))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>{00001}</td>
<td>{0101}</td>
<td>(p_1+p_2p_3(q_2)+p_1p_4p_5(q_2q_3))</td>
</tr>
<tr>
<td>(1,3)</td>
<td>{00110}</td>
<td>{0111}</td>
<td>(p_1p_3+ p_2p_4q_1+ p_1q_5+ p_2p_3p_5(q_1q_4)+ p_1p_3p_4q_2q_5)</td>
</tr>
<tr>
<td>(1,4)</td>
<td>{11010}</td>
<td>{1111}</td>
<td>(p_2+p_3q_5(q_2)+ p_1p_2q_3q_4+ p_4q_2q_5)</td>
</tr>
<tr>
<td>(2,3)</td>
<td>{0000}</td>
<td>{00001}</td>
<td>(p_5+p_3p_4(q_2)+ p_1p_2p_4(q_3q_5))</td>
</tr>
<tr>
<td>(2,4)</td>
<td>{00}</td>
<td>{00101}</td>
<td>(p_4+p_3p_5(q_2)+ p_1p_2p_5(q_3q_4))</td>
</tr>
</tbody>
</table>

The reliability expression (RE) for a node pair is realizable from the proposed data format and vice versa. For example, in Table 1 \(RE(1,2)\) has three terms: \(p_1\), \(p_2p_3(q_2)\) and \(p_2p_4p_5(q_2q_3)\); corresponding to three paths \((k=1,2,3)\). The \(p\) terms appearing outside of the parenthesis for each path are taken from the PL entries while \(p\) and \(q\) terms inside parenthesis are taken from DSR and DSQ respectively.

It is proposed to develop a model library for frequently used network segments. This will help a user to carry out detailed network reliability analysis of large networks with less time and efforts. Next, two proposed operators Union and Intersection, used in later discussions, are presented.

### 2.2 The Intersection (AND) Operator

Intersection of \(RE_1\) and \(RE_2\) is evaluated by intersecting each path of \(RE_1\) with each path of \(RE_2\) as following:

1. **Verifying if the intersection results in simple path:** On intersection of two simple paths, a simple path is formed if there is not more than one link in common in the paths. In case there are at least two nodes in common, a simple path is obtained if number of links is exactly one less than number of nodes in the resulted path. These two conditions are verified easily using AND (\(*\)) operator.
2. **Get disjoint set entries for the simple paths:** by application of OR operator (+) on disjoint terms of two input paths.
3. **Remove disjoint terms having both \(p\) & \(q\) terms for any link:** if a disjoint set is containing both reliability (either in PL or DSR) and unreliability (in DSQ) terms for same links, it will produce non-zero result on application of \(*\) operator and deleted.
4. **Remove repeated \(p\) entries of a link in path and disjoint set entry:** the links present in both DSR and respective PL are removed by replacing DSR with \(\{\sim PL\} * DSR\).
5. **Arrange paths in increasing cardinality**

In the above procedure steps 3 & 4 are not required if the two input pathsets are independent (from different sections). The procedure is explained by following example:
Example 1

Let the intersection operator is applied on two reliability expressions as:

\[ RE = RE(1,4) \times RE(2,3) \]

\( RE(1,4) \) and \( RE(2,3) \) are given in Table 1. These have 4 and 3 paths respectively. On execution of step 1 & 2, \( RE \) has 8 simple paths (in place of \( 4 \times 3 = 12 \) paths) and associated disjoint terms as following:

\[ RE = [p_1 p_3 p_5 + p_1 p_4 p_5 (q_3)] + [p_2 p_3 p_4 (q_1 + p_1 q_5)] + [p_2 p_4 p_5 (q_1 q_3 + p_1 q_3 q_5)] + [p_1 p_2 p_4 (q_1 q_3 q_4 + p_1 q_3 q_4 q_5)] + [p_2 p_3 p_5 (q_1 q_4)] + [p_1 p_3 p_4 (q_2 q_5)] \]

Next, disjoint terms containing both \( q \) and \( p \) terms, as underlined above, are removed.

\[ RE = [p_1 p_3 p_5 + p_1 p_4 p_5 (q_3)] + [p_2 p_3 p_4 (q_1 + p_1 q_5)] + [p_2 p_4 p_5 (q_1 q_3)] + [p_1 p_3 p_4 (q_1 q_4)] + [p_1 p_3 p_4 (q_2 q_5)] \]

Finally, repeated \( p \) terms, as underlined above, are removed and \( RE \) is arranged in increasing cardinality order of paths. It gives \( RE \) as:

\[ RE = p_1 p_3 p_5 + p_1 p_4 p_5 (q_3) + p_1 p_2 p_5 (q_3 q_4) + p_2 p_3 p_4 (q_1 + p_1 q_5) + p_2 p_4 p_5 (q_1 q_3) + p_1 p_2 p_4 (q_3 q_5) + p_2 p_3 p_5 (q_1 q_4) + p_1 p_3 p_4 (q_2 q_5) \]

The distinct advantages of the proposed intersection operator are:

- Early identification of loop formation (redundant paths) saves execution time as it does not evaluate disjoint terms for non contributing paths.
- It finds out redundant paths, which are the loops, by applying simple and fast binary operators on input \( PL \) and \( PN \).
- It does not require the time taking process of comparing each path with previous paths and generating disjointing terms. It evaluates reliability expression from input reliability expressions only.

2.3 The Union (OR) Operator

Unionization of \( RE_1 \) and \( RE_2 \) as \( RE \) is evaluated by moving a path and associated disjoint terms from top of one of the inputs \( RE_1 \) and \( RE_2 \) to bottom of resultant \( RE \). The path is chosen on lower cardinality basis. The input \( RE \) from which path is moved remains same while other input \( RE \) is made disjoint with the path moved using Abraham’s method [12]. The procedure of moving paths and disjointing continues until \( RE_1 \) or \( RE_2 \) becomes empty. Then remaining terms of other \( RE \) are moved to the bottom of resultant \( RE \).

The procedure for application of the union operator on two input reliability expressions is explained by an example below.

Example 2

Let the union operator is applied on two \( RE \) as:

\[ RE = RE(1,4) + RE(2,3) \]

where, \( RE(1,4) \) and \( RE(2,3) \) are given in Table 1. The \( RE \) gives reliability that node \( N_1 \) is connected to node \( N_4 \) OR node \( N_2 \) is connected to node \( N_3 \).

Since top path of \( RE(2,3) \), which is \( p_3 \), has lower cardinality than top path of \( RE(1,4) \), it is moved from top of \( RE(2,3) \) to \( RE \), giving:

\[ RE = p_3; \ RE(1,4) = p_1 p_5 + p_2 p_4 (q_1 + p_1 q_5) + p_2 p_3 p_5 (q_1 q_4) + p_1 p_3 p_4 (q_2 q_5); \text{ and} \]

\[ RE(2,3) = p_4 p_5 (q_3) + p_1 p_2 (q_3 q_4 + p_4 q_3 q_5) \]
Since, \( \text{RE}(1,4) \) is made disjoint with moved path \( p_3 \). This deletes the underlined paths and disjoint terms, which have the path \( p_3 \) as subset and remaining paths of \( \text{RE}(1,4) \) gets multiplied with \( q_3 \). It gives:

\[
\text{RE}(1,4) = p_1p_5 \ (q_3) + p_2p_4 \ (q_1q_3+p_1q_3q_5)
\]

Both \( \text{RE}(1,4) \) and \( \text{RE}(2,3) \) have equal cardinality paths at top and same number of total paths. Therefore, a path can be picked up from any of these RE. Let, path is picked up from top of \( \text{RE}(2,3) \) and \( \text{RE}(1,4) \) is made disjoint with it. It gives:

\[
\text{RE} = p_3 + p_4p_5 \ (q_3), \ \text{RE}(1,4) = p_1p_5 \ (q_3q_4) + p_2p_4 \ (q_1q_3q_5+p_1q_3q_5) \quad \text{and} \quad \text{RE}(2,3) = p_1p_2 \ (q_3q_4+p_4q_3q_5)
\]

This process continues and gives final result as:

\[
\text{RE} = p_3 + p_4p_5 \ (q_3) + p_1p_5 \ (q_3q_4) + p_1p_2 \ (q_3q_4q_5+p_4q_3q_5) + p_2p_4 \ (q_1q_3q_5)
\]

Followings are the benefits of the proposed union operator:

Higher cardinality paths of one of the input RE are made disjoint only with lower cardinality paths of other input RE. It does not re-apply disjointing procedure to already disjoint paths of same RE. This results in execution time saving.

The proposed operator gives resultant RE in the order of cardinality.

Next, modification of node pair reliability expressions with modifications in network layout is explained.

### 2.4 Network Layout Modifications

#### M1. Remove a Set of Links

Let a set of links \( \text{SLD} \) is removed from a network. This change in network is equivalent to changing reliability of these links to zero and unreliability to one for any node pair reliability expression. The procedure to modify reliability expression is:

1. **Remove paths containing any deleted link:** Apply AND operator to test whether any path contains any of the link deleted from the network. A non-zero value means the path is to be deleted with associated disjoint terms.

2. **Remove disjoint terms containing any deleted link:** for remaining paths, apply AND operator on DSR values to verify presence of deleted links. A non-zero value means associated DSR and DSQ terms are to be deleted.

3. **Remove unreliability entries corresponding to deleted links:** From remaining DSQ terms, bits corresponding to deleted links are reset to zero using AND operator.

If link removal results in isolation of nodes then these nodes can be removed from network. Removal of links may also result in network partition. However, there is no problem to store data for partitioned networks with the proposed data structure. Such networks will not have any path for node pairs comprising of nodes from different network partitions. This procedure is explained by an example given below.

**Example 3**

Let link \( X_3 \) is removed from network shown in Figure 1. It results in a new network shown in Figure 2. The procedure to modify \( \text{RE}(2,3) \) for network modification is explained below.

Initially, \( \text{RE}(2,3) = p_3 + p_4p_5 \ (q_3) + p_1p_2 \ (q_3q_4+p_4q_3q_5) \)

Removing all paths having link \( p_3 \) makes, \( \text{RE}(2,3) = p_4p_5 \ (q_3) + p_1p_2 \ (q_3q_4+p_4q_3q_5) \).

Removing all disjoint terms having \( p_3 \) doesn’t make any difference as \( p_3 \) is not present in any disjoint term. Now removing \( q_3 \) terms gives modified \( \text{RE}(2,3) \) as:

\[
\text{RE}(2,3) = p_4p_5 + p_1p_2 \ (q_4+p_4q_5)
\]
M2. Remove a Set of Nodes
A network can be modified by deleting a set of nodes SND. Deleting a set of nodes also removes a set of links SLD connected with these nodes from the network. Therefore, the procedure is same as deleting a set of links. The only modification is paths passing through the node can be easily and quickly determined by comparing PN binary string with SND. This procedure for modifying node pair reliability expressions is as follows:
1. Delete paths, which pass through any of the node from SND,
2. For the remaining paths, if any DSR has any of the deleted links then delete respective DS terms, and
3. Reset bits corresponding to deleted links, SLD, from remaining DSQ terms.

Example 4
Let node N4 is removed from network shown in Figure 1. The resulted network is shown in Figure 3. The modification of \( RE(2,3) \) is explained here:
Initially, \( RE(2,3) = p_3 + p_4p_5(q_3) + p_1p_2(q_3q_4+p_4q_3q_5) \)
Since path \( p_4p_5 \) is passing through node \( N_4 \), it is removed, giving:
\( RE(2,3) = p_3 + p_1p_2(q_3q_4+p_4q_3q_5) \)
Next, links removed are \( X_4 \) and \( X_5 \) so disjoint terms having \( p_4 \) or \( p_5 \) are deleted, giving:
\( RE(2,3) = p_3 + p_1p_2(q_3q_4) \)
Next, link terms having \( q_4 \) or \( q_5 \) are removed from expression, giving \( RE \) as:
\( RE(2,3) = p_3 + p_1p_2(q_3) \)

M3. Add link to a new node
Link addition to a new node from one of the node of a network does not change RE for any node pairs. In this case, RE need to be evaluated for new node pairs formed by combination of earlier nodes of the network with the new node.

Let, a new link \( X_L \) is added from node \( N_j \) of a network to a new node \( N_N \). Then RE for an earlier node \( N_i \) paired with \( N_N \) can be determined as:
\[ RE(i,N) = RE(i,j) \times RE(j,N) \] (1)
where,
\[ RE(j,N) = p_L \]
When \( i=j \), eq. (1) is changed to:
\[ RE(i,N) = RE(j,N) \] (2)

Figure 4: Adding New Link \( X_6 \) to New Node \( N_5 \) from Node \( N_4 \) of Figure 1

The procedure for evaluating above equations is very simple. That is reading the \( RE(i,j) \) value and add \( p_L \) to all paths. This can be achieved by setting \( L^{th} \) and \( N^{th} \) bit to one for all link and node binary strings of paths.

Example 5
An example network is shown in Figure 4. It is obtained from network, shown in Figure 1, by adding a new link \( X_6 \) from node \( N_4 \) to a new active node \( N_5 \). New RE are evaluated for node pairs \( (1,5), (2,5), (3,5), \) and \( (4,5) \). Values of \( j, N \) and \( L \) for this example are 4, 5 and 6 respectively.
When \( j=i=4 \),
\[ RE(4,5) = p_6. \]
Let, \( \text{RE}(1,5) \) is to be evaluated. From Eq. (2):
\[
\text{RE}(1,5) = \text{RE}(1,4) \times \text{RE}(4,5) = \text{RE}(1,4) \times \text{p6}
\]
\( \text{RE}(1,5) \) is obtained by adding \( \text{p6} \) to all paths of \( \text{RE}(1,4) \), given in Table 1. This gives:

\[
\text{RE}(1,5) = p1p5p6 + p2p4p6 (q1+p1q5) + p2p3p5p6 (q1q4) + p1p3p4p6 (q2q5)
\]

It can be observed that the proposed procedure evaluates new \( \text{RE} \) from earlier stored \( \text{RE} \) by setting bits in PL and PN entries of the stored \( \text{RE} \) corresponding to the added link and node. It does not require time taking disjoint procedure.

**M4. Join two network segments through a link**

Joining two network segments through a link does not change \( \text{RE} \) for individual network segments. It introduces new \( \text{RE} \) for node pairs in which one node is taken from each network. The evaluation procedure for the new \( \text{RE} \) is similar to the procedure, for adding a new link to a new node, presented in previous section.

Out of the two segments joined together, nodes and links of one segment are numbered after the node and link numbers of the other segment. The change in link and node numbers can be achieved by shifting bits to left.

Let, the new link \( \{X_{L1+1}\} \) is added between node \( N_j \) of network segment -1 and node \( N_k \) of network segment -2. Path sets for node pairs \( (m, n) \), where node \( N_m \) is taken from segment-1 and \( N_n \) is taken from segment-2, can be determined as:

\[
\text{RE}(m,n) = \text{RE}(m,j) \times \{p_{L1+1}\} \times \text{RE}(k,n) \tag{3}
\]

**Example 6**

As shown in Figure 5, two network segments, Seg#1 and Seg#2, are joined through a new link \( X_6 \) between nodes \( N_4 \) of Seg#1 and \( N_5 \) of Seg#2. Let \( \text{RE}(1,8) \), in which node \( N_1 \) is taken from Seg#1 and \( N_8 \) is taken from segment-2, need to be evaluated.

From Eq. (3),
\[
\text{RE}(1,8) = \text{RE}_{seg1}(1,4) \times \{p_{6}\} \times \text{RE}_{seg2}(5,8)
\]

Using procedure for adding a link to new node we get \( \text{RE}(1,5) = \text{RE}_{seg1}(1,4) \times p6 \). Then, \( \text{RE}(1,8) \) is evaluated by applying intersection operator on \( \text{RE}(1,5) \) and \( \text{RE}(5,8) \) considering them independent.

It can be observed that task of joining two networks through a link becomes quite simple with the stored node pair reliability expressions. Moreover, the same network can be duplicated any number of times to create new networks. This procedure uses the intersection operator, which does not require the time consuming disjoint procedure. Besides reliability expressions being intersected are independent therefore testing or modification of disjoint set terms for simultaneous presence reliability and unreliability terms and duplicate presence of reliability terms is not required.

**M5. Add a new link between two nodes of a network**

Link addition between two nodes of a network modifies reliability expressions for all node pairs of the network. This modification is due to possible addition of paths through the added link. The link addition does not remove any of the earlier paths. Therefore, \( \text{RE} \) for
new network are obtained by first evaluating added paths with associated disjoint terms and then applying the union operator for adding these paths to the previous paths.

**Figure 6:** Adding New Link \( X_7 \) to Between \( N_3 \) and \( N_5 \) of Figure 4

Let a new link \( X_L \) is added between nodes \( N_i \) and \( N_j \) of a network. Then, \( RE(m,n) \) for modified network is obtained as:

\[
RE(m,n)_{\text{new}} = RE(m,n)_{\text{old}} + (RE(m,i | G - N_j, N_n) * p_L * RE(j,n | G - N_i, N_m)) + (RE(m,j | G - N_i, N_n) * p_L * RE(i,n | G - N_j, N_m))
\]  
where,

- \( RE(m,n)_{\text{old}} \) – RE for node pair \( (N_m, N_n) \) before modification
- \( RE(m,i | G - N_j, N_n) \) – RE for node pair \( (N_m, N_i) \) after deleting nodes \( N_j \) and \( N_n \)
- \( RE(m,j | G - N_i, N_n) \) – RE for node pair \( (N_m, N_j) \) after deleting nodes \( N_i \) and \( N_n \)
- \( RE(i,n | G - N_j, N_m) \) – RE for node pair \( (N_i, N_n) \) after deleting nodes \( N_j \) and \( N_m \)
- \( p_L \) – Reliability of the added link

Let, \( RE_{\text{add}}(m,n) = RE_{\text{add}}(m,i | G - N_j, N_n) * p_L * RE_{\text{add}}(j,n | G - N_i, N_m) \) + \( RE_{\text{add}}(m,j | G - N_i, N_n) * p_L * RE_{\text{add}}(i,n | G - N_j, N_m) \)  
Then,

\[
RE(m,n)_{\text{new}} = RE(m,n)_{\text{old}} + RE_{\text{add}}(m,n)
\]  

Depending on values of \( m, n, i, \) and \( j \), Eq. (5) gets modified as following:

**Case 1:** \( i \neq m \neq j \neq n \) (none of \( i, j, m \) and \( n \) is equal to any other)

In this case equation (5) remains same.

**Case 2:** \( i = m \) and \( j \neq n \) (one node is common)

\[
RE_{\text{add}}(m,n) = p_L * RE_{\text{add}}(j,n | G - N_i)
\]  

**Case 3:** \( i = n \) and \( j \neq m \) (one node is common)

\[
RE_{\text{add}}(m,n) = p_L * RE_{\text{add}}(j,m | G - N_j)
\]  

**Case 4:** \( j = m \) and \( i \neq n \) (one node is common)

\[
RE_{\text{add}}(m,n) = p_L * RE_{\text{add}}(i,n | G - N_j)
\]  

**Case 5:** \( j = n \) and \( i \neq m \) (one node is common)

\[
RE_{\text{add}}(m,n) = p_L * RE_{\text{add}}(i,m | G - N_j)
\]  

**Case 6:** \( i = m \) and \( j = n, or, i = n \) and \( j = m \) (both nodes are common)

\[
RE(m,n)_{\text{add}} = p_L
\]  

First, \( RE(m,n)_{\text{add}} \) is evaluated using intersection operator on reliability expressions modified for node deletion. Then, \( RE(m,n)_{\text{new}} \) is evaluated by applying union operator on \( RE(m,n)_{\text{add}} \) and \( RE(m,n)_{\text{add}} \). The node deletion procedure removes those paths from \( RE(m,n)_{\text{add}} \) which are already present in the \( RE(m,n)_{\text{add}} \).

**Example 7**

The example network is shown in Figure 6, which is obtained from the network shown in Figure 4 by adding a new link \( X_7 \) between two nodes \( N_3 \) and \( N_5 \). The procedure to modify an RE explained for \( RE(1,4) \).
Table 2: Node Pair Reliability Expressions for Network Shown in Figure 6

<table>
<thead>
<tr>
<th>Node Pair</th>
<th>Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>RE(1,2)</td>
<td>$p_1 + p_2p_3q_1 + p_2p_4p_5(q_1q_3) + p_2p_6p_7(q_1q_3q_5)$</td>
</tr>
<tr>
<td>RE(1,3)</td>
<td>$p_2 + p_1p_3q_2 + p_4p_6p_7(q_2q_5) + p_2q_4q_5$</td>
</tr>
<tr>
<td>RE(1,4)</td>
<td>$p_1p_5 + p_2p_4(q_1 + p_1q_5) + p_1p_6p_7(q_2q_5 + p_2q_4q_5) + p_2p_3p_5q_4 + p_2p_3p_5q_6 + p_2p_6p_7(q_1q_4q_5)$</td>
</tr>
<tr>
<td>RE(1,5)</td>
<td>$p_1p_5 + p_2p_6(q_1 + p_1q_5) + p_2p_6p_7(q_1q_3q_5) + p_2p_3p_5q_4 + p_2p_3p_5q_6 + p_2p_6p_7(q_1q_4q_5)$</td>
</tr>
<tr>
<td>RE(2,3)</td>
<td>$p_2 + p_1p_3q_2 + p_4p_6p_7(q_2q_5 + p_1q_2q_5)$</td>
</tr>
<tr>
<td>RE(2,4)</td>
<td>$p_5 + p_4p_6q_5 + p_1p_2p_4(q_3q_5 + p_6q_3q_5) + p_1p_2p_4p_6(q_3q_5q_6)$</td>
</tr>
<tr>
<td>RE(2,5)</td>
<td>$q_7 + p_5p_7(q_7) + p_3p_4q_7 + p_1p_2p_4p_6(q_3q_5q_6)$</td>
</tr>
<tr>
<td>RE(3,4)</td>
<td>$p_4 + p_3p_5q_4 + p_2p_3p_4q_7 + p_1p_2p_6p_7(q_3q_4q_5)$</td>
</tr>
<tr>
<td>RE(3,5)</td>
<td>$p_3p_7 + p_4p_6p_7(q_3q_5q_6) + p_1p_2p_6p_7(q_3q_4q_5)$</td>
</tr>
<tr>
<td>RE(4,5)</td>
<td>$p_6 + p_5p_7(q_6) + p_3p_4p_6q_5 + p_1p_2p_4p_6(q_3q_5q_6)$</td>
</tr>
</tbody>
</table>

Substituting values of variables as $m=1; n=4; i=3; j=5$ in Eq. (5), it becomes:

RE_add(1,4) = \{RE_old(1,3|G-N_4, N_5) + p_7 \ast RE_old(1,5|G-N_3, N_5) \} + \{RE_old(1,5|G-N_3, N_4) \ast p_7 \ast RE_old(3,4|G-N_1, N_5) \}

First let us evaluate RE1 = RE_old(1,3|G-N_4, N_5) \ast p_7 \ast RE_old(1,5|G-N_3, N_5). RE_old(1,3|G-N_4, N_5) is obtained after modifications for deleting the nodes N_4 and N_5 using the already explained procedure for deleting nodes of the network. It gives:

RE_old(1,3|G-N_4, N_5) = p_2 + p_1p_3 (q_2)

RE_old(1,5|G-N_3, N_5) = p_6

Applying intersection operator gives,

RE1 = RE_add(1,4) = p_2p_6p_7 + p_1p_3p_6p_7 (q_2)

Similarly, RE2 = RE_add(1,5|G-N_3, N_4) \ast p_7 \ast RE_add(3,4|G-N_1, N_5) is evaluated. However, it results in null set of paths since all paths of RE(1,5) pass through N_4.

Therefore,

RE_add(1,4) = RE1 = p_2p_6p_7 + p_1p_3p_6p_7 (q_2)

RE_new(1,4) = RE_add(1,4) \cup RE_add(1,4)

= p_1p_5 + p_2p_4(q_1 + p_1q_5) + p_2p_4p_6(q_1q_4 + p_1q_4q_5) + p_2p_3p_5q_4 + p_2p_3p_5q_6 + p_1p_3p_4p_6(q_2q_5) + p_1p_3p_4p_6(q_2q_5q_4)

Similar to RE_new(1,4), RE_new for other node pairs are evaluated and given in Table 2.

2.5 k-Terminal Reliability Evaluation

k-terminal reliability (KTR) for a network is stated as – the probability that a given set K of k terminals are connected to each other by a path of working edges. From this statement, an equivalent logical reliability expression is intersection of RE for all k*(k-1)/2 possible node pairs formed by the K set of nodes. However, it is found that application of intersection operator to k-1 node pair reliability expressions, having one node in common, gives K-terminal reliability for undirected networks.

The two input RE are pre-processed before applying intersection to improve efficiency of the procedure. The pre-processing moves paths with associated disjoint set terms, which already have the K set of nodes, from both input RE to the resultant K-terminal reliability expression. The procedure for K-terminal reliability evaluation is illustrated by an example below.

Example 8

Let, the problem is to evaluate 4-terminal reliability of nodes {N_1, N_2, N_3 and N_5} for the network shown in Figure 6. The logical expression of desired KTR is:

RE_add(1,2,3) = RE(1,2) \ast RE(1,3) \ast RE(1,5) = RE(1,2,3) \ast RE(1,5) (6)

In the above logical expression, the node N_1 is taken as common node. The desired KTR
can also be evaluated using other expressions. For example, \( RE(1,5) * RE(1,3) * RE(2,5) \) or \( RE(1,2) * RE(2,3) * RE(3,5) \). From eq. (6):

\[
RE(1,2,3) = RE(1,2) * RE(1,3)
\]

The procedure for evaluation of \( RE(1,2,3) \) is:

- Move paths with their disjoint terms from \( RE(1,2) \) and \( RE(1,3) \), which pass through set of nodes \( \{N_1, N_2, N_3\} \). It gives values of the three variables as:
  \[
  RE(1,2) = p_1; \ RE(1,3) = p_2; \ RE(1,2,3) = p_2p_3 (q_1) + p_2p_4p_5 (q_1q_3) + p_1p_3 (q_2) + p_1p_4p_5 (q_2q_3) + p_1p_4p_6p_7(q_2q_3q_5)
  \]
- Apply intersection operator on \( RE(1,2) \) and \( RE(1,3) \) and add resultant paths to \( RE(1,2,3) \). It gives:
  \[
  RE(1,2,3) = p_1p_2 + p_2p_3q_1 + p_2p_4p_5(q_1q_3) + p_2p_4p_6p_7(q_1q_3q_5) + p_1p_3(q_2) + p_1p_4p_5(q_2q_3) + p_1p_4p_6p_7(q_2q_3q_5)
  \]

The procedure for evaluation of \( RE(1,2,3,5) \) is:

- Move paths with their disjoint terms from \( RE(1,2,3) \) and \( RE(1,5) \), which pass through set of nodes \( \{N_1, N_2, N_3, N_5\} \). Current values of the three variables are:
  \[
  RE(1,2,3) = p_1p_2 + p_2p_3q_1 + p_2p_4p_5(q_1q_3) + p_1p_3(q_2) + p_1p_4p_5(q_2q_3)
  \]
  \[
  RE(1,5) = p_1p_7 + p_1p_5p_6(q_7) + p_2p_4p_5(q_1q_3 + p_3q_1q_7 + p_1q_5q_7)
  \]
  \[
  RE(1,2,3,5) = p_2p_4p_6p_7(q_1q_3q_5) + p_1p_4p_6p_7(q_2q_3q_5) + p_2p_3p_7(q_1) + p_2p_4p_5p_7(q_1q_3q_6) + p_2p_3p_5p_6(q_1q_4q_7) + p_1p_3p_4p_6(q_2q_3q_5)
  \]
- Apply intersection operator on \( RE(1,5) \) and \( RE(1,2,3) \) and add resultant paths to \( RE(1,2,3,5) \). It gives \( RE(1,2,3,5) \) as
  \[
  RE(1,2,3,5) = p_1p_2p_7 + p_1p_3p_7q_2 + p_1p_2p_5p_6(q_7) + p_1p_2p_4p_6(q_5q_7) + p_1p_3p_5p_6(q_1q_3q_5) + p_1p_4p_5p_6(q_2q_3q_7) + p_2p_4p_6p_7(q_1q_3q_5) + p_1p_4p_6p_7(q_2q_3q_5) + p_2p_3p_7(q_1) + p_2p_4p_5p_7(q_1q_3q_6) + p_2p_3p_5p_6(q_1q_4q_7) + p_1p_3p_4p_6(q_2q_3q_5)
  \]

It can be observed from the above example that

- Moving paths containing the K set of nodes from the two input RE to resultant RE reduces the computation requirements.
- The procedure evaluates not only \( RE(1,2,3,5) \) but also \( RE(1,2,3) \). A proper sequence of RE terms in the logical expression can provide not only the final result but also other useful intermediate results.
- This procedure requires the intersection operator only. Therefore, time-taking disjointing operations are not required.
- Similar procedures can be used for evaluating other reliability indices, which can be defined by connecting different RE with union and intersection operators providing flexibility to user in defining and evaluating desired reliability indices of a network.
Table 3: Execution Time Taken in Each Network Modification shown in Figure 8

<table>
<thead>
<tr>
<th>NET #</th>
<th>M#</th>
<th>Step Time (sec)</th>
<th>Cum. Time (sec)</th>
<th>NET #</th>
<th>M#</th>
<th>Step Time (sec)</th>
<th>Cum. Time (sec)</th>
<th>NET #</th>
<th>M#</th>
<th>Step Time (sec)</th>
<th>Cum. Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>14</td>
<td>M5</td>
<td>1.4120</td>
<td>1.7930</td>
<td>27</td>
<td>M5</td>
<td>1639.8900</td>
<td>2914.6000</td>
</tr>
<tr>
<td>2</td>
<td>M3</td>
<td>0.0000</td>
<td>0.0000</td>
<td>15</td>
<td>M5</td>
<td>2.5040</td>
<td>4.2970</td>
<td>28</td>
<td>M3</td>
<td>2.2230</td>
<td>2916.8230</td>
</tr>
<tr>
<td>3</td>
<td>M3</td>
<td>0.0100</td>
<td>0.0100</td>
<td>16</td>
<td>M5</td>
<td>3.4950</td>
<td>7.7920</td>
<td>29</td>
<td>M5</td>
<td>4234.0180</td>
<td>7150.8410</td>
</tr>
<tr>
<td>4</td>
<td>M3</td>
<td>0.0100</td>
<td>0.0200</td>
<td>17</td>
<td>M5</td>
<td>5.5880</td>
<td>13.3800</td>
<td>30</td>
<td>M4</td>
<td>29.6330</td>
<td>7180.4740</td>
</tr>
<tr>
<td>5</td>
<td>M3</td>
<td>0.0100</td>
<td>0.0300</td>
<td>18</td>
<td>M5</td>
<td>8.3020</td>
<td>21.6820</td>
<td>31</td>
<td>M3</td>
<td>15.5120</td>
<td>7195.9860</td>
</tr>
<tr>
<td>6</td>
<td>M3</td>
<td>0.0100</td>
<td>0.0400</td>
<td>19</td>
<td>M5</td>
<td>13.1690</td>
<td>34.8510</td>
<td>32</td>
<td>M3</td>
<td>16.0230</td>
<td>7212.0090</td>
</tr>
<tr>
<td>7</td>
<td>M3</td>
<td>0.0310</td>
<td>0.0710</td>
<td>20</td>
<td>M5</td>
<td>22.2820</td>
<td>57.1330</td>
<td>33</td>
<td>M3</td>
<td>15.4520</td>
<td>7227.4610</td>
</tr>
<tr>
<td>8</td>
<td>M3</td>
<td>0.0300</td>
<td>0.1010</td>
<td>21</td>
<td>M5</td>
<td>39.5170</td>
<td>96.6500</td>
<td>34</td>
<td>M3</td>
<td>16.3140</td>
<td>7243.7750</td>
</tr>
<tr>
<td>9</td>
<td>M3</td>
<td>0.0400</td>
<td>0.1410</td>
<td>22</td>
<td>M5</td>
<td>74.8570</td>
<td>171.5070</td>
<td>35</td>
<td>M4</td>
<td>34.5700</td>
<td>7278.3450</td>
</tr>
<tr>
<td>10</td>
<td>M3</td>
<td>0.0500</td>
<td>0.1910</td>
<td>23</td>
<td>M5</td>
<td>143.7360</td>
<td>315.2430</td>
<td>36</td>
<td>M3</td>
<td>22.9930</td>
<td>7301.3380</td>
</tr>
<tr>
<td>11</td>
<td>M3</td>
<td>0.0600</td>
<td>0.2510</td>
<td>24</td>
<td>M5</td>
<td>286.1610</td>
<td>601.4040</td>
<td>37</td>
<td>M4</td>
<td>42.9420</td>
<td>7344.2800</td>
</tr>
<tr>
<td>12</td>
<td>M3</td>
<td>0.0600</td>
<td>0.3110</td>
<td>25</td>
<td>M5</td>
<td>671.8960</td>
<td>1273.3000</td>
<td>38</td>
<td>M3</td>
<td>41.6500</td>
<td>7385.9300</td>
</tr>
<tr>
<td>13</td>
<td>M3</td>
<td>0.0700</td>
<td>0.3810</td>
<td>26</td>
<td>M3</td>
<td>1.4020</td>
<td>1274.7020</td>
<td>39</td>
<td>M3</td>
<td>31.3950</td>
<td>7417.3250</td>
</tr>
</tbody>
</table>

3. Results and Discussion

The proposed modeling approach is applied on a segment of 45 nodes from institute LAN (IIT-KGP-LAN) layout, shown in Figure 7, to evaluate reliability expressions for all node pairs. The network segment is divided into three sections: a) Core Network, b) Hostel Area Network, and c) Academic Area Network.

The cores network, consists of nodes N1 to N3, is connected using full mesh topology. Academic Area Network, consists of nodes N4 to N45, is connected to node N1 using tree topology. Hostel Area Network, consists of nodes N4 to N16, is connected to two nodes N2 and N3. This whole network segment is connected to rest of the LAN using tree topology.

3.1 Node Pair Reliability Evaluation

The node pair reliability expressions are evaluated for all 45 nodes of the network using network modification procedures (M1-M5). The procedure evaluates 39 different
networks using the proposed algorithms for network modifications. The sequence for application of these modification procedures is provided in Figure 8. Multiple applications of similar modification are grouped together for saving space. For example 11*M3 means modification M3 is carried out 11 times. The execution time taken in each modification step and cumulative execution time are provided in Table 3.

3.1.1 Comparison with Abraham’s Method

The proposed approach is developed using the Abraham’s method [12] for disjointing procedure. Therefore, node pair reliability expressions for NET#39 are evaluated and compared with Abraham’s method on the basis of execution time. The reliability values, obtained using the proposed approach and Abraham’s method, are found same. However, execution time depends on lot of factors e.g. computer hardware, programmer’s skills, operating system, coding language, memory, machine state etc. To minimize effects of these factors, same computer (P-III, 256 MB SDRAM, 20 GB HDD), same operating system (Windows XP), same programming language (MATLAB), same data representation and similar code is used as well as the two methods were run consecutively.

Cumulative execution time taken for evaluating all node pair reliability expressions of the IIT-KGP-LAN (NET#39) using the proposed modelling approach is 7471 sec, which is equal to 2 hours, 3 minutes, and 37 seconds. On the other hand, total execution time taken to evaluate the same, by first generating path sets and then disjoint set using Abraham’s method takes 42067 sec, which is equal to 11 hours, 41 minutes, and 8 seconds. The total execution time taken using the proposed modelling approach is about 5.67 times lesser than the conventional Abraham’s method. Moreover, 39 different networks are evaluated in the proposed approach for reliability expressions of all node pairs. These evaluated networks can be used in any other analysis as well.

From the execution results, following observations can be made:

1. The procedure for adding link between nodes (M5) takes more execution time than any other proposed procedure. However, reduction in time is achieved when a network is evaluated from already evaluated network as it applies disjointing procedure only for the added paths with previous paths.

2. Execution time increases with increase in number of nodes and redundancy in the network. More nodes require modifying/generating reliability expressions for more node pairs and increased redundancy increases number of paths and disjoint terms for node pairs. Therefore, it is best to analyze highly redundant networks in small parts. This is the reason behind less execution time using the proposed model as redundancies are taken care when network is small, having 16 active nodes (NET#29), and then rest of the star connected network is added to it.

3. It is observed that adding the academic area of the network, which uses star topology, takes very less time (266 sec). Therefore, other nodes of the network connected using star topology can be added to the network and all the node pair reliability expressions can be generated without taking much time.

3.2 Evaluation of User-Defined Reliability Indices

Let, reliability for each link is assumed to be 0.9 for all calculations. Let us evaluate the probability that both my hostel (N8) and department (N41) are connected to Library (N24).

It is evaluated by applying intersection operator on RE(24,8) and RE(24,40). The resulted reliability value is 0.71449290. Similarly, if we evaluate reliability that any of the hostel or the department is connected to Library then it is evaluated by application of union operator
Network Reliability Evaluation with Changes in Layout

3.2.1 All Terminal Reliability (ATR) evaluation

All terminal reliability (ATR) for complete IIT-KGP-LAN is evaluated by first evaluating ATR for Hostel Area of LAN followed by ATR evaluation for Academic Area of LAN and combining the two expressions in the end.

<table>
<thead>
<tr>
<th>Set of K Nodes</th>
<th>$t_1^*$ (in sec)</th>
<th>Cum. $t_1^*$ (in sec)</th>
<th>$t_2^{**}$ (in sec)</th>
<th>Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 to 3</td>
<td>90.0000</td>
<td>90.000</td>
<td>226.3250</td>
<td>0.98010000</td>
</tr>
<tr>
<td>1 to 4</td>
<td>202.1710</td>
<td>292.171</td>
<td>236.6120</td>
<td>0.97029900</td>
</tr>
<tr>
<td>1 to 5</td>
<td>195.6320</td>
<td>487.803</td>
<td>246.9150</td>
<td>0.96059601</td>
</tr>
<tr>
<td>1 to 6</td>
<td>144.2070</td>
<td>632.010</td>
<td>369.1210</td>
<td>0.95099005</td>
</tr>
<tr>
<td>1 to 7</td>
<td>106.3030</td>
<td>738.313</td>
<td>660.0990</td>
<td>0.941480149</td>
</tr>
<tr>
<td>1 to 8</td>
<td>88.4870</td>
<td>826.800</td>
<td>1477.9350</td>
<td>0.932065348</td>
</tr>
<tr>
<td>1 to 9</td>
<td>93.1440</td>
<td>919.944</td>
<td>4295.8870</td>
<td>0.922744694</td>
</tr>
<tr>
<td>1 to 10</td>
<td>131.1280</td>
<td>1051.072</td>
<td>5817.8150</td>
<td>0.913517247</td>
</tr>
<tr>
<td>1 to 11</td>
<td>235.8990</td>
<td>1286.971</td>
<td>66999.5660</td>
<td>0.904382075</td>
</tr>
<tr>
<td>1 to 12</td>
<td>532.5050</td>
<td>1819.476</td>
<td>-</td>
<td>0.895338254</td>
</tr>
<tr>
<td>1 to 13</td>
<td>1455.3430</td>
<td>3274.819</td>
<td>-</td>
<td>0.886384872</td>
</tr>
<tr>
<td>1 to 14</td>
<td>0.7710</td>
<td>3275.590</td>
<td>-</td>
<td>0.886384872</td>
</tr>
<tr>
<td>1 to 15</td>
<td>4435.3470</td>
<td>7710.937</td>
<td>-</td>
<td>0.877521023</td>
</tr>
<tr>
<td>1 to 16</td>
<td>4394.1770</td>
<td>12105.114</td>
<td>-</td>
<td>0.868745813</td>
</tr>
</tbody>
</table>

* time taken using the proposed method
** time taken in generating disjoint set using Abraham’s Method

ATR of Hostel area network (NET#29) can be evaluated as:

$$\text{ATR(Net#29)} = \text{KTR}(1 \text{ to } 16) = \text{RE}(1,2) \times \text{RE}(1,3) \times \ldots \times \text{RE}(1,16)$$

Above expression is evaluated using the procedure described earlier for KTR evaluation. The intermediate and final results of the evaluation are given in Table 4. The intermediate results give the k-terminal reliability for nodes 1 to k, when RE(1,k) is intersected with RE(1, 2, … k-1). Reliability values are calculated assuming each link’s reliability equal to 0.9. The time taken in each step ($t_1^*$), cumulative time (Cum. $t_1^*$), time taken by Abraham’s method and k-terminal reliability values are shown in the Table 4.

The Academic area network is star connected network therefore its ATR is equal to multiplication of reliability of its links. With 29 links, its reliability becomes $0.9^{29} = 0.047101287$. The two networks for Hostel and Academic area do not have any common link therefore ATR expressions of these two networks are independent of each other and multiplying the two ATR values gives complete network (NET#39) ATR = 0.040919046.

4. Conclusion

The proposed modelling approach has been successfully applied to various networks and found easy to use. It is illustrated with examples that it provides much flexibility to analyst for evaluating different network reliability indices altogether, besides node pair reliability expressions for all nodes. Moreover, it is fast in analyzing changes in the node pair reliability expressions with changes in network layout. The proposed algorithms are quite efficient and store SVI reliability expressions in compact form, by associating disjoint set terms to path set terms. The proposed approach can be made even faster by removing
those paths which contribute negligibly in reliability expressions and restricting number of
nodes for whom node pair reliability expressions are obtained. Further research may be
carried out to implement MVI techniques and using appropriate methodologies for
approximation of reliability expressions.

Acknowledgement: Thanks are due to anonymous referees who helped improve the
original paper. The author would also like to thank R.B. Misra and S.K. Chaturvedi,
Reliability Engineering Centre, IIT Kharagpur for their support and suggestions.

References

1993.
terminal pair reliability of communication networks. Journal of Quality in Maintenance
[6]. Deo N., and M. Medidi. Parallel algorithms for terminal pair reliability. IEEE Trans. on
Program for Complex Structural Reliability Analysis. IEEE Trans. on Reliability April 1988;
Networks with Imperfect Vertices. IEEE Trans. on Reliability June 2007; 56(2): 288-
300.
[10]. Kuo S.Y., Lu S.K. and F.M. Yeh. Determining Terminal-Pair Reliability Based On Edge
Expansion Diagrams Using OBDD. IEEE Trans. on Reliability Sept. 1999; 48(3): 234-
245.
[12]. Abraham J.A. An Improved Algorithm for Network Reliability. IEEE Trans. on

Neeraj Kumar Goyal received his Ph.D. from IIT Kharagpur in Reliability Engineering
in 2006. He received the Bachelor of Engineering (Hons.) degree in Electronics &
Communications Engineering from MREC Jaipur, Rajasthan, India in 2000. He has served
as an Executive in M/s Secure Meters Ltd., Udaipur from 2000 to 2001. He is currently Sr.
Lecturer in Reliability Engineering Centre, Indian Institute of Technology Kharagpur,
India. He is providing consultancy services to national defence and nuclear power
organizations in the area of reliability and safety engineering.