Efficient Algorithms for Reliability Evaluation of General Networks

Mohamed-Larbi Rebaiaia¹² and Daoud Ait-kadi¹²

(1) Interuniversity Research Center on Enterprise Networks, Logistics and Transportation
Laval University, Quebec (QC), Canada
Mohamed-larbi.rebaiaia@cirrelt.ca

(2) Mechanical Engineering Department, Laval University
Laval University, Quebec (QC), Canada
Daoud.ait-kadi@gmc.ulaval.ca

ABSTRACT

Several production systems either for goods or services can be modeled by a network where nodes are production centers, warehouses, distributions and others, and arcs represent the relationship between nodes. Nodes and arcs are often subjected to random failures that may result from several causes. These networks include one or more sources and one or more destinations. Given the stochastic nature of the failure, the reliability and the robustness of the network become an important criteria for safety, economical and environment reasons.

Several methods based on graph theory and stochastic processes are proposed in the literature. The concepts of minimal paths set (MPS) and minimal cuts set (MCS) as well as decomposition techniques based on Bayes' theorem have been widely used. The performance of these methods is greatly affected by network size (number of nodes and arcs) and its density.

Generally, except for special structure of some networks (e.g series, parallel, standby, etc.) there is no mathematical expression based on the reliability of its nodes and its arcs that has been proved compact for representing the expression of the reliability function of any network. This paper attempts to provide solutions to this problem by proposing and testing a unified approach based on MPS/MCS and Binary Decision Diagrams (BDD). This approach is illustrated by several simple examples. A tool has been developed to handle complex networks such as telecommunication networks and other network’s tests published in the literature.

Keywords
Reliability, Networks, Algorithms, MPS, MCS, BDD.
1. INTRODUCTION

The assessment of network’s reliability has been addressed in many papers [1, 9, 10, 11, 12, 13, 14, 18, 20, 24, 25, 26, 35, 39]. Efficient computation techniques have been proposed for the network reliability over a given mission duration. The complexity of network’s reliability algorithms increases with the size of the networks (number of arcs and nodes).

This paper proposes an integrated approach using MPS/MCS and BDD for generating the network structure function which can be used for evaluating the reliability of networks so that the considered system meets its end-to-end service availability objective. A concrete telecommunication is used to testify the applicability of all the algorithms that will be proposed in the following sections and a series of network’s benchmark are also used to show the strength of the algorithms and to compare their performance within those published in the literature.

Most of the methods and algorithms proposed up to now for determining the reliability of networks consider three categories of techniques which are different in the form but are very close in substance. These techniques have been discussed in many publication researches and concern a large number of physical systems such as electric power systems, telecommunication networks [24, 25, 26, 27], traffic and transportation systems, just to name a few. Generally, reliability engineers model the functioning and the physical connectivity of system components by a network. Mathematically, a network is a graph \( G(V, E) \) in which \( V \) represents the components (e.g. devices, computers, routers, etc.) and \( E \) the interconnections (e.g. HF-VHF and microwave transporters [24]. In the sequel, we suppose that graphs and networks and systems are of the same object and can be used to design the same system.

Network reliability analysis consists of evaluating the 2-terminal reliability of networks \( (K \) and all-terminal) [10, 11, 31, 33]. General theory, has discussed extensively two techniques; exact [1, 8, 12, 13, 14, 20, 21, 24, 34] and approximate methods [23]. The exact method uses the concept of MPS/MCS [11, 25, 26, 29, 32, 34, 35]. Determining MCS is essential not only to evaluate the reliability but also to investigate different scenarios in order to find for instance the redundant components that could be replaced to improve the load point reliability and for predicting the risk that a part of a system could fail or not with a certain probability value [26, 38]. Enumerating all MCS may be a preferable way if the number of paths is too huge to be practically enumerated than the number of cuts. Examples of this kind of preferences is the 2x100 lattice which has \( 2^{99} \) paths and just 10000 cuts [20], and complete network with 10 nodes contains 109601 minimal paths and just 256 cuts [26]. In existing algorithms [11, 24, 35], minimal paths are deduced from the graph using simple and systematic recursive algorithms that guarantee the generated paths set to be minimal. The enumeration of MCS is more problematic because they need advanced mathematics, set theory and matrices manipulation. Many algorithms have been published in the literature and some of them are implemented in commercial tools. Enumeration appears to be the most computationally efficient. An initiative of solution has been proposed in [21, 22] where author presented a method for generating MPS directly from MCS, or vice-versa. It starts with the inversion of the reliability expression accomplished by a recursive method combining a 2-step application of De Morgan's theorem. Yan et al. [34] presented a recursive labelling algorithm for determining all MCS in a directed network, using an approach adapted from dynamic programming algorithms. The algorithm produces all MCS, and uses comparison logic to eliminate redundant cuts. This algorithm is an enumeration technique derived from the approach of Jensen & Bellmore [14] and follows an extension of Tsukiyama et al. [32] to improve the computational efficiency and space requirements of the algorithm. Jasmon and Kai [13] use an algorithm which proceeds by deducting first the link cuts set from node cuts set and, second the basic minimal paths using network decomposition. In addition to the enumeration of cuts set directly, it is possible to obtain them from the inversion of minimal paths [22], Shier and Whited [29]. In such topic, one of the best algorithms is proposed by Al-Ghanim [3] which is based on a heuristic programming algorithm for generating all MPS and Cuts set. Recently, Rebaiaia and Ait-Kadi [24] propose an elegant and fast algorithm for enumerating MPS using a modified DFS technique [30]. The
procedure uses each discovered path to generate new MPS from subpaths. The above procedure is repeated until all minimal paths (MP) are found. The algorithm didn’t produce any redundant MPS. Furthermore, they extended their work with theoretical proofs and the usage of sophisticated techniques for dynamic data structures manipulation of complex networks.

The paper is structured as follows. Section 2 presents some related preliminaries. Section 3 details some algorithms for determining the MPS, MCS and BDD. Section 4 and section 5 present some experienced applications using a series of benchmarks and a real telecommunication system. The paper concludes the paper in section 6.

2. Backgrounds

2.1. Stochastic Graph

A probabilistic graph $G = (V, E)$ is a finite set $V$ of nodes and a finite set $E$ of incidence relations on the nodes called edges. The edges are considered as transferring a commodity between nodes with a probability $p$ called reliability. They may be directed or undirected and are weighted by their existence probabilities. The graph in such case, models a physical network, which represents a linked set of components giving services. The reliability of networks is defined as the probability that systems (networks) will perform their intended function without failure over a given period of time. Figure 1 shows an example of an undirected graph where node 1 and node 6 are respectively initial node and terminal node.

For a specified set of nodes $K \subseteq V$ of $G$, we denote the $K$-terminal reliability of $G$ by $R(G_K)$. When $|K| = 2$, $R(G_K)$ is called a 2-terminal (or terminal-pair) reliability which defines the probability of connecting $(s, t)$ a source node with a target node [25, 31, 33]. A success set, is a minimal set of the edges of $G$ such that the vertices in $K$ are connected; the set is minimal so that deletion of any edges causes the vertices in $K$ to be disconnected and this will invalidate the evaluation of the reliability. Topologically, a success set is a minimal tree of $G$ covering all vertices in $K$. The computation of the $K$-terminal reliability of a graph may require efficient algorithms. One such solution can be derived directly from the topology of the network by constructing a new parallel-series network using MPS of the original network such that each minimal path constitutes a branch of the parallel-series graph. Then, a characteristic expression $\Phi(x)$ called structure function is derived from the disjoint expressions of paths terms, and from which the reliability is evaluated after applying Boolean simplification processing [25].

![Figure 1: A probabilistic weighted graph with six nodes (1, 2, 3, 4, 5, 6) and nine undirected edges (a, b, ..., i).](image)

2.2. Multicomponent Reliability

Let $f(x)$ be the failure probability density function of any component and $t$ the time period beginning from time zero. The reliability of a component may be expressed as

$$R(t) = \int_t^{\infty} f(x)dx$$  (1)
When a system is composed by \( m \) identical components disposed in series or \( n \) identical components disposed in parallel, the mathematical expressions of their reliability function are respectively

\[
R_x(t) = \prod_{i=1}^{m} \int_{t}^{\infty} f_i(x) \, dx
\]

(2)

\[
R_p(t) = 1 - \prod_{i=1}^{n} \left( 1 - \int_{t}^{\infty} f_i(x) \, dx \right)
\]

(3)

It is simple to combine series with parallel and parallel with series configurations. Their expressions are respectively

\[
R_{x,p}(t) = 1 - \prod_{i=1}^{m} \left( 1 - \prod_{j=1}^{n} \int_{t}^{\infty} f_{ij}(x) \, dx \right)
\]

(4)

\[
R_{p,x}(t) = \prod_{i=1}^{n} \left( 1 - \prod_{j=1}^{m} \left( 1 - \int_{t}^{\infty} f_{ij}(x) \, dx \right) \right)
\]

(5)

When systems are complex and does not possess a well-defined structure, it would be necessary to determine their reliability using methods which proceed by successive transformations. An interesting introduction to such methods has been published in [25] and [26].

2.3 Structure Function

Consider a system composed of \( m \) components numbered from 1 to \( m \). Each of these components may be in functioning state or in failed state with a probability \( p_i \) respectively a probability \( q_i \). Let \( x_i \) be the state component and \( x \) the state vector, they can be defined as follows:

\[
x_i(t) = \begin{cases} 
1 & \text{if the component } i \text{ (node/edge) is UP at time } t \\
0 & \text{if the component } i \text{ (node/edge) is down}
\end{cases}
\]

\( x = (x_1, x_2, ..., x_m) \) is the state vector of a system \( S \) of order \( m \) such that \( x \in \Omega = \{0,1\}^m \) the state space of the system.

Mathematically, any system can be represented by its structure function which is an application function taking its value in the Boolean domain, such that,

\[
\Phi(x) = \begin{cases} 
1 & \text{if the system is functioning when the state vector is } x \\
0 & \text{if the system has failed when the state vector is } x
\end{cases}
\]

In this paper, we consider only systems that are coherent represented by structure functions that are non-decreasing [4].

2.4 Minimal Paths set and Minimal Cuts set

Let \( (s, t) \) be a fixed initial and terminal nodes in a graph representing the system. A minimal path is any path composed by a series of successive edges linking \( s \) to \( t \) and such that if any one of such edges is removed from the path, the link between \( s \) to \( t \) is broken. Respectively, a minimal cut is composed by a set of edges such that if one edge is removed from the cut, the link from \( s \) to \( t \) still functioning. Note that if we remove any minimal cut edge, the link from \( s \) to \( t \) is broken.
Thus, a minimal paths set (MPS) respectively a minimal cuts set (MCS) are composed by all minimal paths respectively all minimal cuts in the graph.

If we suppose that a system is composed by a set MPS = \{P_1, P_2, ..., P_p\} and a set MCS = \{C_1, C_2, ..., C_r\} its structure function can be expressed by:

\[
\Phi(x) = \max_{i \in MPS} \min_{j \in P_i} x_i = \min_{i \in MCS} \max_{j \in C_j} x_i
\]

and if \(E\{\Phi(X)\}\) is expected mathematical expression, the reliability of a system is computed according to the following formula:

\[
R(G) = E\{\Phi(X)\} = Pr(\Phi(X) = 1) = \sum_{X \in \Omega} \Phi(X)Pr(X = x)
\]

and such that the probability \(Pr(X = x)\) is determined by \(p_i = Pr(X = 1)\) and \(q_i = Pr(X = 0)\) = 1 - \(p_i\).

Note that the formula in Equation (6) means that the structural function of any complex system is equivalent to the structural function after transforming such system to a parallel-series or series-parallel one. Figure 2 shows clearly such transformation.

2.5 Reliability Evaluation

After enumerating MPS and MCS, the reliability evaluation needs the development of the symbolic expression in terms of the probability of the various components being operational/non-operational, for that calculation, Equation (6) and Equation (7) can be used. But, it is not always easy to do so, especially for complex systems. Fortunately, we can find in the literature many other techniques and algorithms to calculating the reliability [12, 16, 17, 33, 34].

2.5.1 Illustrative example

Consider a directed bridge network as represented in Figure 2 (a). The MPS (Figure 2 (c)), MCS (Figure 2 (d)), the structure function and the reliability expression are respectively:

MPS: \(p_1 = \{x_1, x_4\}; p_2 = \{x_2, x_3\} \text{ and } p_3 = \{x_1, x_3, x_5\}\).

MCS: \(c_1 = \{x_1, x_2\}; c_2 = \{x_1, x_5\}; c_3 = \{x_4, x_3\} \text{ and } c_4 = \{x_2, x_3, x_4\}\).

Structure function:

\[
\Phi(X(t)) = 1 - (1 - x_1 x_4)(1 - x_2 x_5)(1 - x_1 x_3 x_5) \text{ (Determined from MPS), and the reliability is,}
\]

\[R(t) = Pr(\Phi(X(t)) = 1) = r_1 r_4 + r_2 r_5 + r_1 r_3 r_5 - r_1 r_2 r_4 r_5 - r_1 r_3 r_4 r_5 - r_1 r_2 r_3 r_5 + r_1 r_2 r_3 r_4 r_5\]

![Figure 2. a) System structure, b) Reliability bloc diagram, c) Reliability structure based on MPS. d) Reliability structure based on MCS.](image-url)
3. Algorithm for determining Network Structure Function

We have just say above that the structure function plays a very important role for calculating the reliability of any system provided that it can be modelled as a graph, and their determination is not as easy as we thought, especially for large networks. Also, to make the structure function expression possible, we have to enumerating all the minimal paths/minimal cuts. For doing that, we can apply any one of the many algorithms published in the literature [25]. But, the problem with MPS and MCS determination is related to their size that can grows exponentially and reaches millions of paths/cuts for graphs that are supposed to be simple such as complete, grid and lattice graphs that represent the most adequate structures to represent systems similar to those of telecommunication and transportation networks [26, 20, 35].

There is another interesting mathematical tool called BDD for binary decision diagram that can be used for determining the system structure function and possibly from which it can be possible to generating both MPS and MCS [6, 7, 8, 26, 27, 35].

3.1 Binary Decision Diagram

Binary Decision Diagrams are the state-of-art data structure for manipulating and simplifying large Boolean expressions such that of analogic and digital function, and specifically those that cannot be handled with traditional techniques such as Table truth and Karnaugh map. Bryant [6, 7] was the first to binary decision diagrams for symbolic verification of integrated circuits. The problem with BDD representation despite their effectiveness is their exponential growing size due to wrong ordering declaration of variables. Bollig et al. [5] demonstrate that improving the Variable Ordering of OBDD is NP-Complete. Ruddell [28] first used an algorithm based on dynamic programming techniques to reduce the size of the BDD.

In engineering, Coudert and Madre [8] and Rauzy [27] applied first, BDDs for evaluating networks reliability. Figure 3 shows the BDD relative to the network presented in Figure 2, and its representation in the computer memory.

Figure 3. (a) : ROBDD corresponding to the network of figure 2.(a) and (b) : its representation code in memory similar to table in figure 5 (right).

The implementation and manipulation of BDD algorithms is composed by three procedures, restrict, apply and ite. The are used for simplifying and reducing Boolean functions. For example, the ite for (If _ Then _ Else) function for a Boolean expression is expressed by
Efficient Algorithms for Reliability Evaluation of General Networks

M-L. Rebaiaia and D. Ait-kadi

Frontiers in Science and Engineering - Vol. 6 - n° 1 - 2016
An International Journal Edited by The Hassan II Academy of Science and Technology

\[ f = \text{ite}(x,F_1,F_2) = x \cdot F_1 + \neg x \cdot F_2; \] with \( F_1 = f_{x=1} \) and \( F_2 = f_{x=0} \).

The following pseudo-code gives the \text{ite} function.

\begin{verbatim}
Function ite(f, g, h)
  if f = 0 then
    Return h;
  else if f = 1 then
    Return g;
  else if (g = 1) \& (h = 0) then
    Return f;
  else if g = h then
    Return g;
  else if \exists \text{computed-table entry (f, g, h)} then
    Return H;
  end if
  \( x_k \leftarrow \text{top variable of f, g, h}; \)
  H \leftarrow \text{new non-terminal node with label}
  \( x_k; \)
  then
    H \leftarrow \text{ite}(f | x_k = 1, g | x_k = 1, h | x_k = 1);
  else
    H \leftarrow \text{ite}(f | x_k = 0, g | x_k = 0, h | x_k = 0);
  end if
  Reduce H;
  Add entry (f, g, h, H) to computed-table;
  Return H;
end.
\end{verbatim}

Another important algorithm incorporated in BDD is the \text{apply} procedure which can manipulating Boolean operator like conjunction, disjunction and complementation Boolean operators and the objective is to mix individual formulas composing a Boolean expression. An example of composition using \text{apply} procedure is shown in Figure 4.

![Figure 4. APPLY procedure mixing the minimal paths formulas of the Network in Figure 2.](image)

3.1.1 Algorithm for BDD procedure

The representation and the simplification of a Boolean expression proceeds in 4-steps:

- Construct the binary decision tree (BDT) associated with the graph formula.
- Transform the BDT to a BDD by applying the following rules:
  - Merging equivalent leaves of a binary decision tree.
  - Merging isomorphic nodes.
  - Elimination of redundant tests.
- Transform the BDD to OBDD by just a wise choice on variables and then to obtain a Reduced OBDD.
- Reduced OBDD can be reduced to a ROBDD by repeatedly eliminating in a bottom-up fashion, any instances of duplicate and redundant nodes. If two nodes are duplicates, one of them is removed and all of its incoming pointers are redirected to its duplicate. If a node is redundant, it is removed and all incoming pointers are redirected to its just one child.
M-L. Rebaiaia and D. Ait-kadi

Efficient Algorithms for Reliability Evaluation of General Networks

In the following, we propose three categories of algorithms for determining the structure function of any graph network. The first algorithm try to determine the MPS, the second the MCS and the third is more general and can be applied on the fly for determining the function structure and at the same time the reliability.

3.2 Algorithm for MPS determination

Let \( G = (V, E) \) be the graph network modelling a system and suppose that the couple \((s, t)\) represents the initial and the terminal nodes of the graph.

It is recognized that the best technique for enumerating the MPS is to use a procedure based on the well-known recursive algorithm called depth first search (DFS) which search successive nodes and edges composing a path from \( s \) to \( t \). When reaching the last node (target node), the algorithm turned back to try to concatenate another part of a path to create a new path. As the procedure is recursive, it continues to explore new paths using other directions until it is not possible to generate other paths.

The algorithm proceeds as follows

**Function** stack \( S = \text{pathDFS}(G, v, z) \)

- setLabel(v, VISITED)
- S.push(v)
- if \( v = z \)
  - return S.elements()
- for all \( e \) in \( G.\text{incidentEdges}(v) \)
  - if getLabel(e) = unvisited
    - \( w \leftarrow \text{opposite}(v,e) \)
  - if getLabel(w) = unvisited
    - S.push(e)
    - pathDFS(G, w, z)
  - else
    - S.pop(v)
- end

**Program** Main()

- **Input**: A connected graph with node set, edge set, a source node, and a sink node.
- Declaring dynamics vectors and stacks (put in them zeros)
- Declaring initial and terminal nodes \((v, z)\)
- Do While .true.
  - pathDFS(G, v, z)
  - if “the last minimal path have been encountered”
    - return .false.
- enddo
- **Output**: All MPS in the graph.
3.2.1 Example

Using the graph in figure 1 and applying the MPS algorithm, we determine all the minimal paths for directed and undirected graphs.

a. Directed graph

![Graph](image)

Figure 5. A 6-node, 9-link example network (directed)

Table 1. The MPS corresponding to the network in Figure 5.

<table>
<thead>
<tr>
<th>#</th>
<th>Minimal Pathset</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1, 3, 5, 7, 9</td>
</tr>
<tr>
<td>2</td>
<td>1, 3, 5, 8</td>
</tr>
<tr>
<td>3</td>
<td>1, 3, 6, 9</td>
</tr>
<tr>
<td>4</td>
<td>1, 4, 7, 9</td>
</tr>
<tr>
<td>5</td>
<td>1, 4, 8</td>
</tr>
<tr>
<td>6</td>
<td>2, 5, 7, 9</td>
</tr>
<tr>
<td>7</td>
<td>2, 5, 8</td>
</tr>
<tr>
<td>8</td>
<td>2, 6, 9</td>
</tr>
</tbody>
</table>

b. Undirected graph (Figure 1)

Because the graph is undirected we must duplicating the edges as two arcs in both opposite directions and then to rename them differently. Table 2 gives such renaming.

![Graph](image)

Table 2. Adjacent matrix of the network in Figure 1.

<table>
<thead>
<tr>
<th>Nodes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>7</td>
<td></td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>14</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>17</td>
<td>18</td>
</tr>
</tbody>
</table>
Table 3. The MPS corresponding to the network in Figure 1.

<table>
<thead>
<tr>
<th>#</th>
<th>Minimal Pathset</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1, 4, 8, 12, 16</td>
</tr>
<tr>
<td>2</td>
<td>1, 4, 8, 13</td>
</tr>
<tr>
<td>3</td>
<td>1, 4, 9, 15, 13</td>
</tr>
<tr>
<td>4</td>
<td>1, 4, 9, 16</td>
</tr>
<tr>
<td>5</td>
<td>1, 5, 11, 9, 16</td>
</tr>
<tr>
<td>6</td>
<td>1, 5, 12, 16</td>
</tr>
<tr>
<td>7</td>
<td>1, 5, 13</td>
</tr>
<tr>
<td>8</td>
<td>2, 7, 5, 12, 16</td>
</tr>
<tr>
<td>9</td>
<td>2, 7, 5, 13</td>
</tr>
<tr>
<td>10</td>
<td>2, 8, 12, 16</td>
</tr>
<tr>
<td>11</td>
<td>2, 8, 13</td>
</tr>
<tr>
<td>12</td>
<td>2, 9, 15, 13</td>
</tr>
<tr>
<td>13</td>
<td>2, 9, 16</td>
</tr>
</tbody>
</table>

3.3 Algorithm for MCS determination

We propose three algorithms for enumeration all minimal cuts set in a graph. They are as follows:

3.3.1 Algorithm for directed graphs

This algorithm is similar of the one published by Yan et al [34], but instead using set theory as a support to data structure, it uses lists objects. It is as follows:

Algorithm DirectedMCS

Let \( A_j, OA_j, p, k, S_k \) are successively set of input and output arcs to and from a node \( j \), \( p \) is a label, \( k \) is node name and \( S_k \).

Step 1. Set \( X_1 = \{1\}, Y_1 = \{2, 3, \ldots, n\} \) // \( n \) : is the number of nodes

- Node 1 is marked
- Select the successor and predecessor of node \( j \) (say \( j' \)) or \( (OA_j, IA_j) \) and define
- \( X_i = X_{i-1} + \{j\} \), \( Y_i = Y_{i-1} - \{j\} \) and determine the \( C_j^p \) using the formula:

\[
C_j^p = C_j^p \cup OA_j - C_j^p \cap IA_j, p = \{1, 2, 3, \ldots, S_k \} \text{ for all } k.
\]

Step 2.
- \( if \ Y_i = \{n\} \) and all nodes of \( X_i = \{1, 2, 3, \ldots, n - 1\} \) have been marked then return and STOP.
  \( else \) GOTO Step 1

end.

3.3.2 Algorithm for undirected graphs

There are many algorithms published in the literature used for determining MCS, but few of them are simple to be used especially the following one:
Algorithm UndirectedMCS
Let \( X \) be the set of nodes including the initial node and such that these nodes are connected between themselves, and let \( G \ast i \) be the graph the transformation of the graph \( G \) in which the node \( i \) has been merged into \( X \) by deleting any edge connecting \( i \) and \( X \).
Let \((s, t)\) be the initial and the terminal nodes of the graph, and suppose that \( i = s \) (initialization); hash table = nil; MCS = nil; (hash table and MCS are empty);
Begin
If \( G(E, V) \) is empty (there isn’t at least two nodes linked by an edge)
Return and STOP
else if \((i = t)\) return
  else \( G = G \ast i, X = X + \{i\} \);
  If \((X\) is present in the hash table) return;
  else Add \( X \) to hash table; end
  Add \( X \) to MCS;
  For all \( i \) adjacent to \( X \) do
    Call Undirected MCS
  end
end.

3.3.3 Algorithm for determining MCS from MPS

Locks [22] and Shier and Whited [29] propose a technique for obtaining MCS by complementing MPS using DeMorgan’s laws [19].

If we transpose the idea of generating MCS by inversion but instead on Boolean formulas, we applied such idea directly on the structure of the BDD (e.g. Figure 3.3). The procedure to deduce the MCS is a depth first search algorithm; it works on the graph using data information’s taken from matrix of the Figure 3 (b). It can be presented as follows:

Procedure Generation of MCS
- Place the squared node on top of a stack 1 //*
  records DFS visits to ROBDD nodes */.
- Place the squared node on top of a stack 2 //*
  records cut’s nodes.
- Place on the top of the stack 1 all the ascending nodes of the top variable in the stack.
- Place the node top of the stack 1 on top of the stack 2, if the edge (link) is dotted.
- Continue until the variable reach the root node.
- If so, a cut has been found. Write the content of the stack 2 as a line of a matrix. Remove top variable from stack 1 and from stack 2.
- Continue the procedure until stack 1 is empty.
- Apply the filtering process by removing all the redundant paths (cuts) using the matrix of paths (cuts)
- Display MSC Matrix.
end
3.4 Algorithm for determining the reliability from BDD data structure

Network reliability is calculated the following algorithm:

```
Procedure Reliability_Evaluation(F, G)
if (F == 0) return 1  //* Boolean value 1 (one) */
else if (F == 1) return 0  //* Boolean value 0 (zero) */
    else if (computed-table has entry {F, P_F})
        return P_F
    else P_F = Prob(F1) + P(x) * (Prob(F2)-Prob(F1))
end
end
Insert_computed_table ({F, P_F})
return P_F
end
```

The following BDD structure shows how the values corresponding to each level are communicated to the upper level until the root.

```
Figure 6. Solution given by the earlier algorithm
```

4. Experimental results

The proposed algorithms and procedures have been implemented in MatLab 8 and Java Jdk 1.6. A communicating interface has been written to render easy data and results transfer between MatLab system and Java packages running under jGrasp a graphical tool written in Java JDK. The operating system is 32 bits and 2038 MO of Windows Vista of Microsoft. The machine is an HP notebook PC with an Intel(R) core (TM) 2 Duo processor of 1.67. The benchmark networks in figure 9 were used and the results are shown in Table 4 and Table 5. All the networks are 2-terminal and they have been used in different publication papers. We can remark from table 5, that the value of execution time is interesting despite the fact that the performance of the machine characteristics is not high. The importance of this work shows the efficiency of the algorithms. Note that no comparison was made with another implementation but they can be compared for example with the results of Lin et al. [20].
The following BDD structure shows how the values corresponding to each level are communicated to the upper level until the root.

Figure 6. Solution given by the earlier algorithm

4. Experimental results

The proposed algorithms and procedures have been implemented in MatLab 8 and Java Jdk 1.6. A communicating interface has been written to render easy data and results transfer between MatLab system and Java packages running under jGrasp a graphical tool written in Java JDK. The operating system is 32 bits and 2038 MO of Windows Vista of Microsoft. The machine is an HP notebook PC with an Intel(R) core (TM) 2 Duo processor of 1.67. The benchmark networks in figure 9 were used and the results are shown in Table 4 and Table 5. All the networks are 2-terminal and they have been used in different publication papers. We can remark from table 5, that the value of execution time is interesting despite the fact that the performance of the machine characteristics is not high. The importance of this work shows the efficiency of the algorithms. Note that no comparison was made with another implementation but they can be compared for example with the results of Lin et al. [20].

Figure 7. Benchmark networks

Table 4. Benchmark results for 2-terminal networks

<table>
<thead>
<tr>
<th>Networks</th>
<th>MPS</th>
<th>MCS</th>
<th>Time(sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>8</td>
<td>12</td>
<td>0.075591</td>
</tr>
<tr>
<td>B</td>
<td>18</td>
<td>110</td>
<td>0.282727</td>
</tr>
<tr>
<td>C</td>
<td>115</td>
<td>85</td>
<td>313.17</td>
</tr>
<tr>
<td>D</td>
<td>33</td>
<td>72</td>
<td>11.29</td>
</tr>
<tr>
<td>E</td>
<td>35</td>
<td>30</td>
<td>0.046</td>
</tr>
<tr>
<td>F</td>
<td>114</td>
<td>562</td>
<td>21236.06</td>
</tr>
<tr>
<td>G</td>
<td>10</td>
<td>959</td>
<td>818.80</td>
</tr>
<tr>
<td>H</td>
<td>29</td>
<td>29</td>
<td>0.708676</td>
</tr>
<tr>
<td>I</td>
<td>25</td>
<td>20</td>
<td>0.332611</td>
</tr>
<tr>
<td>J</td>
<td>13</td>
<td>21</td>
<td>4.059842</td>
</tr>
<tr>
<td>K</td>
<td>44</td>
<td>528</td>
<td>2572.03</td>
</tr>
<tr>
<td>L</td>
<td>6</td>
<td>23</td>
<td>0.226500</td>
</tr>
<tr>
<td>M</td>
<td>36</td>
<td>96</td>
<td>11.166</td>
</tr>
<tr>
<td>N</td>
<td>100</td>
<td>16</td>
<td>8.3297</td>
</tr>
<tr>
<td>O</td>
<td>98</td>
<td>105</td>
<td>283.38</td>
</tr>
<tr>
<td>P</td>
<td>5</td>
<td>16</td>
<td>0.768533</td>
</tr>
<tr>
<td>Q</td>
<td>13</td>
<td>9</td>
<td>0.945836</td>
</tr>
</tbody>
</table>
5. CASE STUDY- A Radio communication network

To illustrate the performance of the algorithms presented in sections 3 and 4, we propose a practical application to a case study problem of undirected regional radio communication network showed in Figure 90 and 10. The system composed of equipment’s is scattered across a wide geographic area. It consists of a set of mobile and portable transmitter-receivers deserved by a network of fixed equipment’s located in Canada. There are two master site in operation 24 hours a day and a third one in standby, which is used in case of urgency, and more than 150 base stations used to transmit the signal generated through the microphone to portable and mobile equipment. A master site consists of core and exit routers, WAN and LAN switches, controllers and some operative computers plus others monitoring and dispatching hardware/software systems such as gateway routers, AEB, PBX, dispatching consoles Elite, and so. The radio sites is equipped with one or two antennas for broadband coverage on which is terminated 4 to 8 transmitter-receiver transponders (Tx / Rx). The transponders are connected to each antenna via filtration equipment of type Multicoupler. The multicouplers form a chain of multicoupling able to accept other transponders in expansion. The range of the base station depends on its power, antenna system, terrain, carrier transporter (e.g. T1 or E1) and environmental conditions (see. [24, 25, 26] for more details).
Figure 9. The reduced architecture of the radio communication network presented in Figure 10
(The case study)

Figure 10. A regional radio communication network

Table 6. The reliability values on each node of the network

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
<th>V</th>
<th>VI</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.99992</td>
<td>0.99992</td>
<td>0.99992</td>
<td>0.99990</td>
<td>0.99990</td>
<td>0.99996</td>
</tr>
<tr>
<td>B</td>
<td>0.99993</td>
<td>0.99987</td>
<td>0.99996</td>
<td>0.99918</td>
<td>0.99989</td>
<td>0.99980</td>
</tr>
<tr>
<td>C</td>
<td>0.99987</td>
<td>0.99993</td>
<td>0.99996</td>
<td>0.99991</td>
<td>0.99992</td>
<td>0.99993</td>
</tr>
<tr>
<td>D</td>
<td>0.99999</td>
<td>0.99997</td>
<td>0.99995</td>
<td>0.99987</td>
<td>0.99970</td>
<td>0.99993</td>
</tr>
<tr>
<td>E</td>
<td>0.99998</td>
<td>0.99992</td>
<td>0.99990</td>
<td>0.99996</td>
<td>0.99996</td>
<td>0.99992</td>
</tr>
<tr>
<td>F</td>
<td>0.99992</td>
<td>0.99993</td>
<td>0.99987</td>
<td>0.99996</td>
<td>0.99988</td>
<td>0.99996</td>
</tr>
<tr>
<td>G</td>
<td>0.99992</td>
<td>0.99992</td>
<td>0.99995</td>
<td>0.99986</td>
<td>0.99987</td>
<td>0.99987</td>
</tr>
<tr>
<td>H</td>
<td>0.99992</td>
<td>0.99987</td>
<td>0.99996</td>
<td>0.99987</td>
<td>0.99992</td>
<td>0.99996</td>
</tr>
<tr>
<td>I</td>
<td>0.99993</td>
<td>0.99992</td>
<td>0.99984</td>
<td>0.99996</td>
<td>0.99993</td>
<td>0.99987</td>
</tr>
<tr>
<td>J</td>
<td>0.99993</td>
<td>0.99990</td>
<td>0.99996</td>
<td>0.99986</td>
<td>0.99992</td>
<td>0.99996</td>
</tr>
<tr>
<td>K</td>
<td>0.99997</td>
<td>0.99992</td>
<td>0.99987</td>
<td>0.99998</td>
<td>0.99993</td>
<td>0.99996</td>
</tr>
<tr>
<td>L</td>
<td>0.99992</td>
<td>0.99995</td>
<td>0.99992</td>
<td>0.99993</td>
<td>0.99975</td>
<td>0.99996</td>
</tr>
<tr>
<td>M</td>
<td>0.99992</td>
<td>0.99987</td>
<td>0.99992</td>
<td>0.99988</td>
<td>0.99986</td>
<td>0.99990</td>
</tr>
<tr>
<td>N</td>
<td>0.99991</td>
<td>0.99997</td>
<td>0.99992</td>
<td>0.99984</td>
<td>0.99993</td>
<td>0.99992</td>
</tr>
<tr>
<td>O</td>
<td>0.99992</td>
<td>0.99992</td>
<td>0.99992</td>
<td>0.99996</td>
<td>0.99992</td>
<td>0.99999</td>
</tr>
<tr>
<td>P</td>
<td>0.99997</td>
<td>0.99996</td>
<td>0.99937</td>
<td>0.99996</td>
<td>0.99987</td>
<td>0.99999</td>
</tr>
<tr>
<td>Q</td>
<td>0.99945</td>
<td>0.99980</td>
<td>0.99990</td>
<td>0.99980</td>
<td>0.9992</td>
<td>0.99999</td>
</tr>
<tr>
<td>R</td>
<td>0.99987</td>
<td>0.99992</td>
<td>0.99980</td>
<td>0.99960</td>
<td>0.99993</td>
<td>0.99987</td>
</tr>
<tr>
<td>S</td>
<td>0.99993</td>
<td>0.99987</td>
<td>0.99987</td>
<td>0.99987</td>
<td>0.99992</td>
<td>0.99993</td>
</tr>
<tr>
<td>T</td>
<td>0.99992</td>
<td>0.9999</td>
<td>0.99990</td>
<td>0.99986</td>
<td>0.99988</td>
<td>0.99998</td>
</tr>
</tbody>
</table>
### Table 7. The reliability of each microwave link between two nodes.

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
<th>V</th>
<th>VI</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>B</td>
<td>0.99992</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>C</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>D</td>
<td>0.99999</td>
<td>0.99998</td>
<td>0.99998</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>E</td>
<td>0.99999</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99998</td>
<td>0.99998</td>
</tr>
<tr>
<td>F</td>
<td>0.99997</td>
<td>0.99997</td>
<td>0.99999</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99997</td>
</tr>
<tr>
<td>G</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>H</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>I</td>
<td>0.99999</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>J</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99998</td>
</tr>
<tr>
<td>K</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>L</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>M</td>
<td>0.99998</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>N</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>O</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>P</td>
<td>0.99996</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>Q</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>R</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>S</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>T</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
<tr>
<td>V</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.99999</td>
</tr>
</tbody>
</table>

### Table 8. (s, t) - Reliability joining any two nodes (dimension = 156 x 156 links).

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>II</th>
<th>III</th>
<th>IV</th>
<th>V</th>
<th>VI</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.99853</td>
<td>0.99759</td>
<td>0.99623</td>
<td>0.99684</td>
<td>0.99684</td>
<td>0.9985</td>
</tr>
<tr>
<td>B</td>
<td>0.99805</td>
<td>0.99788</td>
<td>0.99753</td>
<td>0.99785</td>
<td>0.99786</td>
<td>0.9966</td>
</tr>
<tr>
<td>C</td>
<td>0.99931</td>
<td>0.99717</td>
<td>0.9971</td>
<td>0.9973</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>0.99914</td>
<td>0.9991</td>
<td>0.9992</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td>0.99697</td>
<td>0.9969</td>
<td>0.9971</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td></td>
<td>0.99841</td>
<td>0.9984</td>
<td>0.9985</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td>0.9984</td>
<td>0.9985</td>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td>0.9985</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
6. CONCLUSION

The network reliability evaluation is an NP-complete problem. The complexity increases with the density of the networks. For small size networks, minimal paths set and minimal cuts can be easily used efficiently for reliability calculation.

This paper introduced a series of algorithms for evaluating the reliability of general networks regardless of their orientation. The proposed algorithms were validated on several test networks. They perform efficiently in terms of computation time and robustness.

Some computer tools were developed and used for computing the reliability of an extensive telecommunication network. They have also been used to identify key components of the network to improve network performance indicators in terms of security, availability and serviceability, both in design and in operations. These tools are currently used to determine the spare parts inventory to maximize network availability while respecting budgetary constraints and the service levels required by the different network’s users.

REFERENCES

Efficient Algorithms for Reliability Evaluation of General Networks

M-L. Rebaiaia and D. Ait-kadi

Frontiers in Science and Engineering - Vol. 6 - n° 1 - 2016
An International Journal Edited by The Hassan II Academy of Science and Technology