Adjacency Matrix based method to compute the node connectivity of a Computer Communication Network

Kamalesh V. N1 and S K Srivatsa2

1Department of Computer Science and Engineering, Sathyabama University, Chennai, India
2 St. Joseph College of Engineering, Chennai, India

Abstract
Survivability of a computer communication network is the ability of a network to provide continuous service in the presence of link or node failures. It is an essential and considerable concern in the design of high speed communication network topologies. The connectivity number of a network is the graph theoretical metric to measure survivability of the communication network. Given a network and a positive integer k, few heuristics exist in literature to verify whether the given network is k connected or not. This paper presents a method to compute the connectivity number k of a given computer communication network.

Keywords: Adjacency Matrix, Node Connectivity, Computation, Network.

1. Introduction

The topological design of a network assigns the links and link capacities for connecting the network nodes. This is a critical phase of network synthesis, partly because the routing, flow control, and other behavioral design algorithms rest largely on the given network topology. The topological design has also several performance and economic implications. The node locations, link connections, and link reliability are security considerations. Some networks may be required to provide more than one distinct path for each node pair, thereby resulting in a minimum degree of connectivity between the nodes. The topological design algorithms must select links and discrete link capacities within several constraints, such as connectivity, message transmission delay, cost, and network traffic. Perhaps the most difficult aspect of topological design is to permit possible future expansion of the network. Such expansion may require increased link capabilities or additional links or nodes [6,16].

The topological design of computer communication network can be modeled by a graph. This fact has been universally accepted and used by computer scientists and engineers. Moreover, practically it as been demonstrated that graph theory is a very powerful mathematical tool for design and analysis of computer network. When the network requirements are expressed in terms of graph theoretical parameters, the problem of analysis and design of networks becomes finding a graph G satisfying some specified requirement [8,9].

One of the basic principles of network design as discussed by Bermond and Peyrat is maximum survivability of the network. The network must continue to work in case of vertex or edge failures. Different notions of fault tolerance exist, the simplest one corresponding to connectivity of the graph, that is, the minimum number of vertices which must be deleted in order to destroy all paths between a pair of vertices. The maximum connectivity is desirable since it corresponds to not only the maximum fault tolerance of the network but also the maximum number of internally disjoint paths between any two distinct vertices [6,8,9].

A fundamental problem in network design is to construct a minimum cost network that satisfies some specified connectivity requirements between pair of nodes. One of the most well suited problems in this framework is the survivable network design problem, where we are given a computer communication network with costs on its edges, and a connectivity requirement rij for each pair i,j of nodes. The goal is to find a minimum cost subset of edges that ensures that there exist rij disjoint paths for each pair i, j of nodes, where all rij {0,k}, for some integer k. We will refer to these problems as k-connectivity of a computer communication network [15]. Few methods for generating k-connected networks are proposed in the literature [2,12,13,14]. Once a potential network N is generated by any one of the existing methods [2,12,13,14] and claims that the connectivity number of the generated network N is k, then a few heuristics exist in literature to verify that the generated network is k connected or not [3,4,5]. All these heuristics [3,4,5]
verifies the claim on the connectivity number of the given network. To compute the connectivity number of the given network one has to run the heuristics iteratively i number of times, where 0<i<d, ‘d’ is the degree of the network graph. In this paper, we present a method to compute the connectivity number κ of a given network directly in a single iteration.

2. Proposed Method

This section presents the proposed method to compute the node connectivity κ of a given network graph. Consider the given network graph N (V, E). The nodes of the network graph are numbered using [18]. Subsequently it is essential to find nodes disjoint paths between each pair of nodes. For this we need to preserve the adjacency relationship between every pair of nodes. One way of preserving the adjacency relationship among the nodes of the network is through the data structures called adjacency matrix. Hence in this work we propose to represent the network using the adjacency matrix. Hence compute the adjacency matrix and the degree d of the network graph N(V,E). Corresponding to these d+1 nodes of L1 create counters A[1], A[2] …A[d+1] and initialize them to zero respectively. Form the set L2 = {d+2,d+3,...,n} consisting of remaining {n-(d+1)} nodes of the given network graph N(V,E).

1. For every i ∈ L1, check for the adjacency of node i with every other node. If the nodes i, j ∈ L1 are adjacent, then increment A[i] and A[j] by one. If the node i ∈ L1 and node u ∈ L2 are adjacent, then increment the counters of A[i] and B[k] by one respectively. If node i ∈ L is adjacent to more than one node of L2, then check whether those nodes of L2 are adjacent. If yes, increment only the counters corresponding to the node of L1, i.e., increment A[i] only by one. If not, then increment the counters corresponding to the nodes of L1 and L2 by one.

2. After considering all the n nodes of the given network, sort the values stored in the counters. The lowest value of the sorted list is the connectivity number κ of the given network graph N (V, E).

Algorithm end.

2.1 Illustration 1

To compute the connectivity number κ of the given network graph N(V,E).

Consider the given network graph as shown in figure 1.
The adjacency matrix and the degree $d$ of the given network graph $N(V,E)$ is computed.

<table>
<thead>
<tr>
<th>Nodes</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Adjacency matrix of the given graph

For the given network graph, the degree $d$ is 2. Form the set $L_1$ consisting of first $d+1$ nodes, i.e., $L_1={1,2,3}$. Corresponding to the nodes of $L_1$, create counters $A[1], A[2]$ and $A[3]$ respectively and initialize them to zero, i.e.,


Form the set $L_2$ consisting of remaining nodes of the given network $N(V,E)$, i.e., $L_2={4,5}$. Corresponding to the nodes of $L_2$, create counters $B[4], B[5]$ and initialize them to zero, i.e., $B[4]=0, B[5]=0$.

Starting from node 1, check for the adjacency of node 1 with every other node. Here, node 1 is adjacent to node 2. Increment the counters $A[1]$ and $A[2]$ by 1 respectively, i.e.,

$A[1] = 0+1 = 1$
$A[2] = 0+1 = 1$

Next, node 1 is adjacent to node 3. Hence, increment $A[1]$ and $A[3]$ by 1 respectively, i.e.,

$A[1] = 1+1 = 2$
$A[3] = 0+1 = 1$

Check for the adjacency of node 2. Here, node 2 is adjacent to node 3. Therefore increment $A[2]$ and $A[3]$ by 1, i.e.,

$A[2] = 1+1 = 2$
$A[3] = 1+1 = 2$

Check for the adjacency of node 3. Node 3 is adjacent to node 4, and node 4 $\notin L_1$.

Further node 3 is adjacent to node 5 also and node 5 $\notin L_2$. In such case, we have to check whether node 4 is adjacent to node 5. Here, in the given network $N(V,E)$, node 4 is adjacent to node 5. Hence, increment the counter corresponding to node 3 only by 1 twice, once corresponding to the adjacency of node 3 with node 4 and once corresponding to the adjacency of node 3 with node 5 i.e.,

$A[3] = 2+1=3$ (Corresponding to Node 4)
$A[3] = 3+1=4$ (Corresponding to Node 5)

This process avoids counting the paths through node 3 more than once. Check for the adjacency of node 4. Here, node 4 is adjacent to node 5. Therefore increment the counters $B[4]$ and $B[5]$ by 1, i.e.,

$B[4] = 0+1 = 1$
$B[5] = 0+1 = 1$

After checking all the nodes for their adjacencies, now check the value stored in the counters corresponding to the nodes of the network.

$A[1] = 2$
$B[4] = 1$
$B[5] = 1$

The minimum amongst the values stored in the counters corresponding to the nodes of the network is 1; hence the network is 1-connected.

2.2 Illustration 2

To compute the connectivity number $\kappa$ of the given network graph $N(V,E)$.

Consider the given network graph as shown in figure 2
The adjacency matrix and the degree d of the given network graph \(N(V,E)\) is computed. For the given network graph the degree \(d\) is 3.

Form the set \(L_1\) consisting of first \(d+1\) nodes, i.e., \(L_1=\{1,2,3,4\}\), corresponding to the nodes of \(L_1\), create counters \(A[1], A[2], A[3]\) and \(A[4]\) respectively and initialize them to zero, i.e., \(A[1]=0, A[2]=0, A[3]=0, A[4]=0\).

Form the set \(L_2\) consisting of remaining nodes of the given network \(N(V,E)\), i.e., \(L_2=\{5,6\}\). Corresponding to the nodes of \(L_2\), create counters \(B[5], B[6]\) and initialize them to zero, i.e., \(B[5]=0, B[6]=0\).

Starting from node 1, check for the adjacency of node 1 with every other node.

Here, node 1 is adjacent to node 2. Increment the counters \(A[1]\) and \(A[2]\) by 1 respectively, i.e.,
\[
A[1] = 0+1 = 1 \\
A[2] = 0+1 = 1
\]

Node 1 is also adjacent to Node 4. Hence, increment \(A[1]\) and \(A[4]\) by 1 respectively, i.e.,
\[
A[1] = 1+1 = 2 \\
A[4] = 0+1 = 1
\]

Node1 is also adjacent to node6. Hence increment \(A[1]\) and \(B[6]\) by 1, i.e.,
\[
A[1] = 2+1 = 3 \\
B[6] = 0+1 = 1
\]

Check for the adjacency of node 2. Here, node 2 is adjacent to node 3. Therefore increment \(A[2]\) and \(A[3]\) by 1, i.e.,
\[
A[2] = 1+1 = 2 \\
A[3] = 0+1 = 1
\]

Node 2 is also adjacent to node5. Hence increment \(A[2]\) and \(B[5]\) by 1, i.e.,
\[
A[2] = 2+1 = 3 \\
B[5] = 0+1 = 1
\]

Check for the adjacency of node 3. Here node 3 is adjacent to node 4. Hence increment \(A[3]\) and \(A[4]\) by 1, i.e.,
\[
A[3] = 1+1 = 2 \\
A[4] = 1+1 = 2
\]

Node 3 is also adjacent to node5. Hence increment \(A[3]\) and \(B[5]\) by 1, i.e.,
\[
A[3] = 2+1 = 3 \\
B[5] = 1+1 = 2
\]

Check for the adjacency of node4. Node 4 is adjacent to node6. Hence increment \(A[4]\) and \(B[6]\) by 1, i.e.,
\[
A[4] = 2+1 = 3 \\
B[6] = 1+1 = 2
\]

Check for the adjacency of node5. Node5 is adjacent to node6. Hence increment \(B[5]\) and \(B[6]\) by 1, i.e.,
\[
B[5] = 2+1 = 3 \\
B[6] = 2+1 = 3
\]

After checking all the nodes for their adjacencies, now check the values stored in the counters corresponding to the nodes of the given network.
\[
A[1] = 3 \\
B[5] = 3 \\
\]

The minimum amongst the values stored in the counters corresponding to all the nodes of the given network is 3, hence the network is 3-connected.
3. Conclusions

One of the basic principles in the topological design of computer communication network is to achieve maximum survivability. It is the ability of the network to provide continuous service in the presence of link or node failure. It is an essential and considerable concern in the design of high speed communication network topologies. The connectivity number of a network in this graph theoretical metric is to measure the survivability of the communication network. This paper presents a method to compute the connectivity number of a given network directly in a single iteration. All the existing methods [3,4,5] only verify whether the given network is $k$-connected or not. To compute the connectivity number of a given network using the methods [3,4,5], one has to run the heuristics iteratively ‘i’ number of times, where $0 < i < d$, ‘d’ is the design of network. The main strength of this paper is that the method computes the connectivity number $k$ in a single iteration.

References


Prof. Kamalesh V. N received the Bachelor of Science degree from University of Mysore, India. Subsequently, he received Master of Science in Mathematics degree and Master of Technology in Computer Science & Technology degree from University of Mysore, India. He secured 14th rank in Bachelor of Science and 4th rank in Master of Science from University of Mysore. Further, he was National Merit Scholar and subject Scholar in Mathematics. He is working as Head, Department of Computer Science & Engineering at JSS Academy of Technical Education, Bangalore, affiliated to Visvesvaraya Technological University, Belgaum, Karnataka, India. He has taught around fifteen different courses at both undergraduate and post graduate level in mathematics and Computer science and engineering. His current research activities pertain to computer networks, Design and Analysis of algorithms, Graph theory and Combinatorics, Finite Automata and Formal Languages. His paper entitled “On the assignment of node number in a computer communication network” was awarded certificate of merit at World Congress on Engineering and Computer Science 2008 organized by International Association of Engineers at UC Berkeley, San Francisco, USA. He is currently research candidate pursuing Ph.D degree from Sathyabama University, Chennai, India.

Dr. S.K. Srivatsa received the Bachelor of Electronics and Telecommunication Engineering degree from Jadavpur University, Calcutta, India. Master’s degree in Electrical Communication Engineering and Ph.D from the Indian Institute of Science, Bangalore, India. He was a Professor of Electronics Engineering...
in Anna University, Chennai, India. He was a Research Associate at Indian Institute of Science. Presently he is working as senior Professor in department of Instrumentation & Control Engineering at St. Joseph College of Engineering, Chennai, India. He has taught around twenty different courses at undergraduate and post graduate level. His current research activities pertain to computer networks, Design and Analysis of algorithms, coding Theory and Artificial Intelligence & Robotics.