Computer Science and Engineering

University of Nevada, Reno



CPE 400 Computer Communication Networks

Fall 2017

Self-healing Mesh Network Project

Team Members:

Rony Calderon, Zachary Johnson, and Logan Long

December 6th, 2017

Self-healing Mesh Network

**Abstract:**

Communication networks are the backbone of the modern Internet infrastructure. The Internet is a network of networks responsible for sending and guiding data through different routing mechanisms from a source to a destination. Each packet of information sent requires the analysis of the packets source and destination addresses for proper traversal through the various routing points to reach its destination. Routing points can be connected in a mesh configuration that allows traversal through the network from node to node. The issue with mesh network schemes arise when an important network node is no longer reachable, and the remaining nodes are forced to re-route packets through a different channel. Additionally, if there is no alternate channel, then the remaining nodes are set to a stand-still and the connection from nodes can be terminated until the offending node is repaired. The solution to this type of issue is to produce a self-healing mesh network that can detect the absence of a routing node and redirect packet traffic to an alternate route. This approach would ensure a network’s robustness and operation in a dynamic network.

**Introduction:**

The modern Internet consists of many networks configured and organized in many schemes. One of the schemes used is a mesh network where the network consists of various nodes interconnected to each other. There are many variants of the mesh network topology, but two of the most popular methods are a full mesh topology and a partial mesh topology. A network that consists of a full mesh network topology requires each network node to be interconnected to every other node in the network. Each node is physically connected to every other node and a full network mesh is generally utilized as a backbone to some systems where connection faults are tolerable such as telecommunication networks. Company or intranet mesh networks may not employ a full mesh network due to their complexity but generally deploy partial mesh networks that fit the application for the company. Some issues with partial mesh networks are node faults and severed links between nodal points that play a critical role in maintaining the integrity of the system. If such a node or dependent link were to fail, then part or most of the network may no longer be reachable. For this reason, self-healing mesh networks play a vital role in maintaining the structural integrity of a mesh network system when nodes or links to a node fail unexpectedly.

**Full Mesh and Partial Mesh Networks:**

The advantages to the full mesh network approach include a more fail-safe and robustly fault-tolerant network. In addition, computers and network devices can switch between multiple redundant connections if a fault is detected. This leads to the robustness of the full mesh network as alternate paths are available in the event of a node fault. The disadvantages of the full mesh network approach begin with the complex wiring and connections which can grow dramatically. Moreover, total installation costs for such a network can become extremely expensive as additional nodes require additional links to be created to keep the full mesh network in proper working order. Maintenance of the full mesh network is also complex and grows in complexity as more nodes and links are added to the network. For a full mesh network with number of nodes, the amount of physical connections needed to maintain the full mesh network integrity starting with node 1 is shown in Table 1.

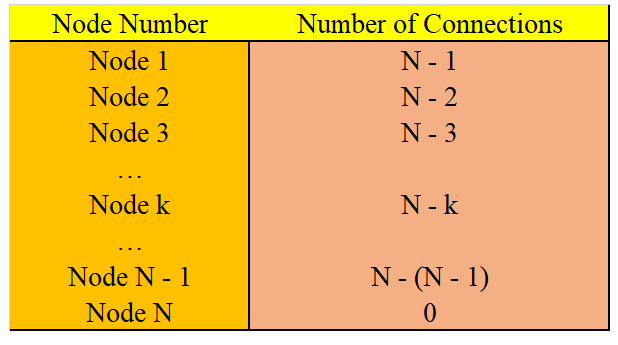


Table 1: Number of connections needed for a full mesh network

To find the total amount of connections required for a full mesh network topology, we need to sum up each of the connections needed. So, we let be the number of total connections needed for a full mesh topology. Then,

Reversing the terms to make the calculation simpler, we get:

This calculation tells us that as the number of network nodes grows, so too does its amount of connections and its complexity in maintaining the full mesh network. Figure 1 shows a simple full mesh network which consists of six nodes and its attributed links to every other node. For six nodes as shown in Figure 1, the full mesh network topology requires a total of 15 connection links.

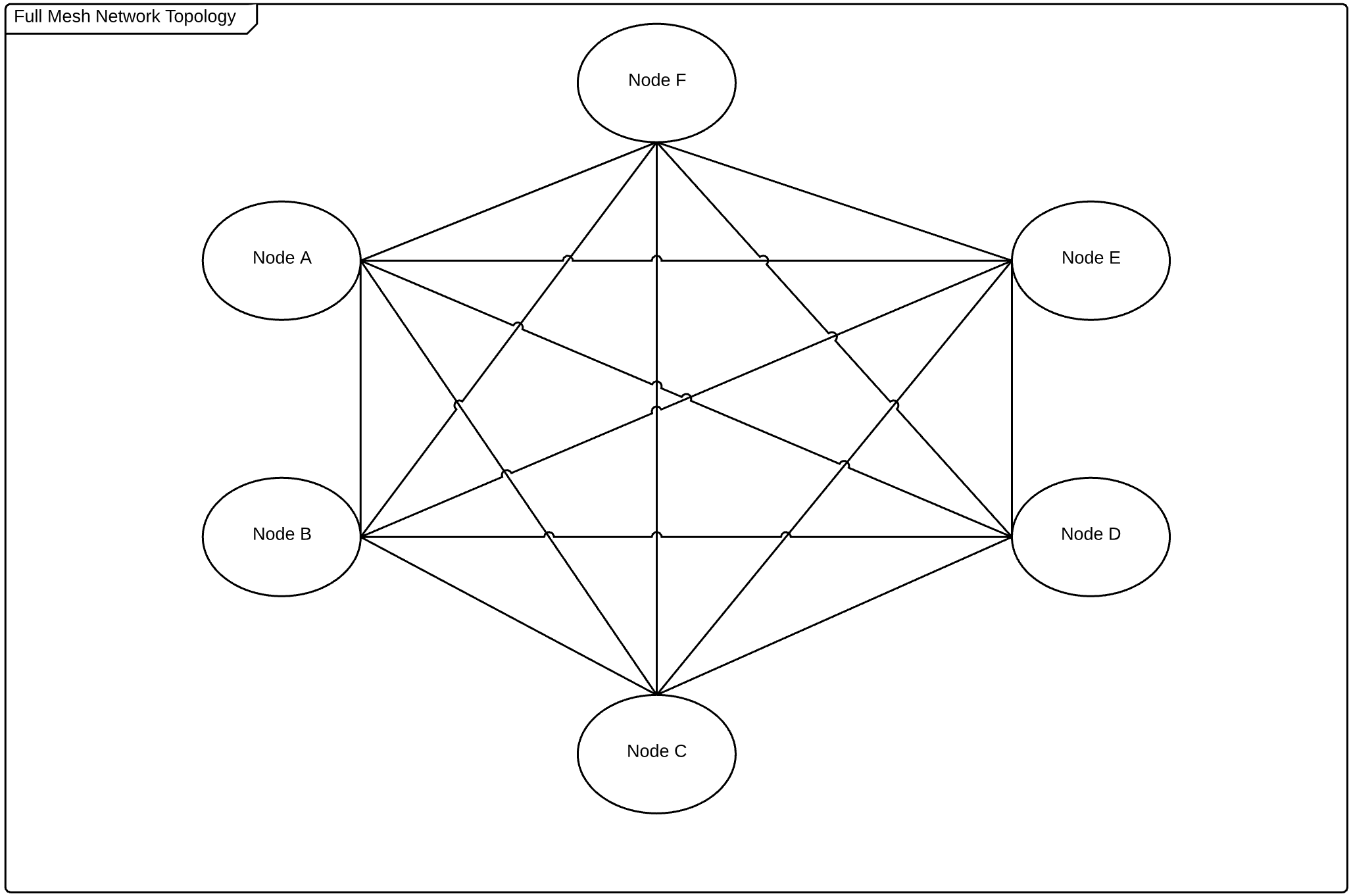


Figure 1: Full mesh network topology

The second common approach to a mesh network is known as a partial mesh network. In a partial mesh network scheme, nodes can connect to more than one node present in the network using a point-to-point method. The point-to-point method takes advantage of the redundancy provided by a full mesh network topology but decreases the complexity of the network and the amount of total connections needed to maintain the partial mesh network integrity. The key difference between a full mesh network and a partial mesh network is that nodes in a full mesh network have a direct connection to all other nodes present in the network while the partial mesh network may only have some connections with other clients. Therefore, there are certain nodes in a partial mesh network that are dependent on their neighbor nodes in order to traverse entirely throughout the network. Figure 2 shows a partial mesh network of the same six nodes provided in the full mesh network depiction.

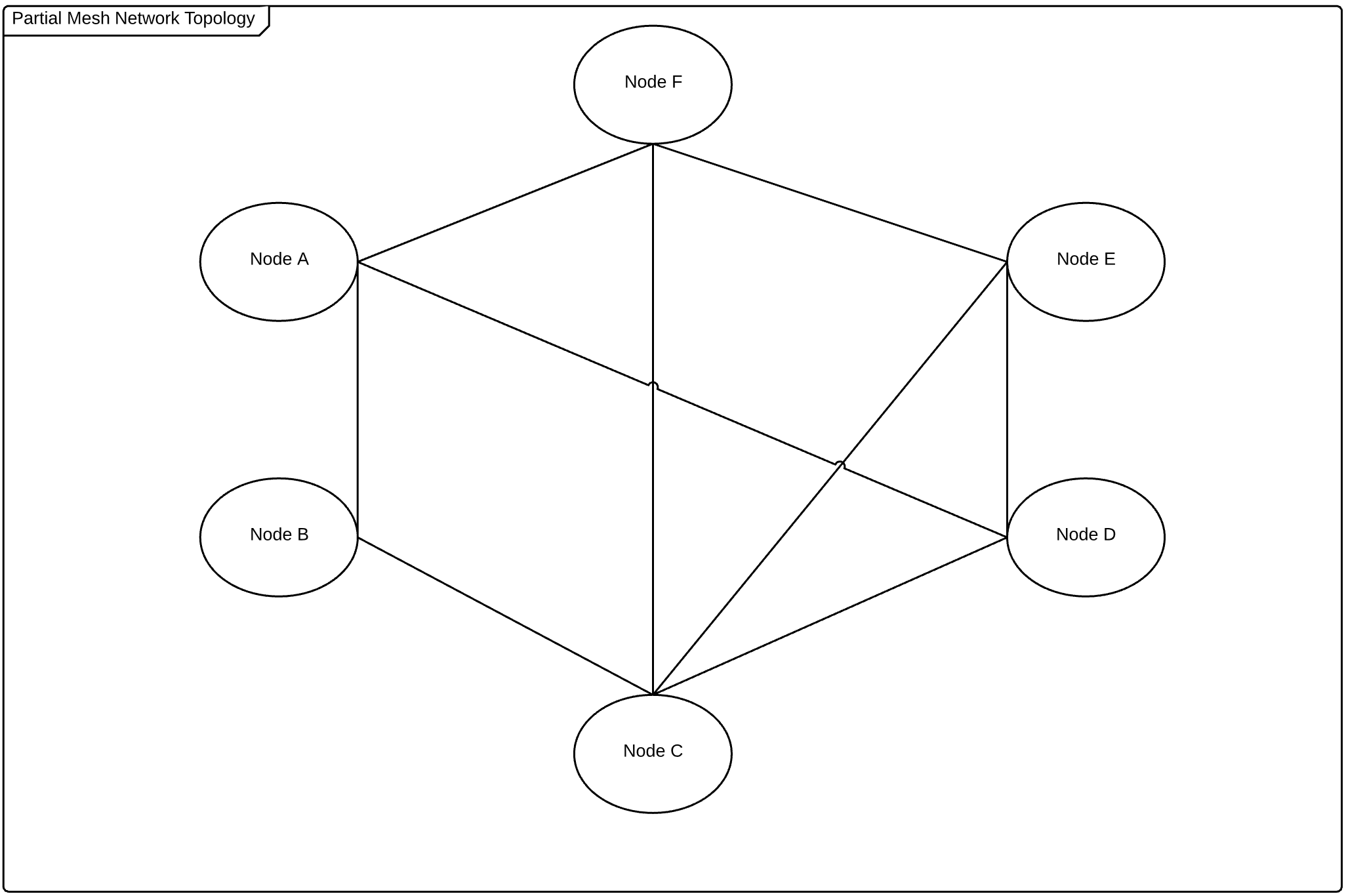


Figure 2: Partial mesh network

The partial mesh network scheme works to take advantage of the robustness of the full mesh network. A single link from node to node , can be broken and the network would still hold its partial mesh network integrity. The full mesh and partial mesh networks assume that the links between the nodes are already made, but what would occur if a critical link between nodes were to fail? In short, the system would cease in its communication between nodes that were dependent on the failed links until the links are repaired. Another approach would be to have the nodes affected to search the network for an additional node to ensure traversal on the network around the failed node. This approach is known as a self-healing approach for mesh networks.

**Self-Healing Mesh Network Theory:**

There are various reasons why a self-healing network is more desirable such as a more robust and reliable network for long term networks. There are also several reasons why they are not desirable due to the complexity involved in initialization and maintenance. Due to these benefits and complexities, a devised method has been created to help satisfy both concerns. The methods set forth are not expressed to be the most efficient or reliable but constitute a theoretical approach in finding every single individual network path for a mesh network. This approach is usually done with link-state algorithm such as Dijkstra’s shown in pseudo code in Figure 3.

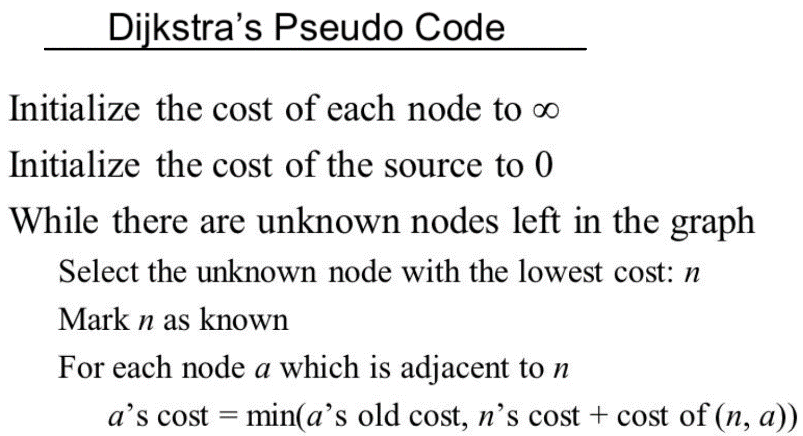


Figure 3: Pseudo code for Dijkstra's algorithm for a network of n amount of nodes.

The primary reason to locating each individual network link is to satisfy the network need for shortest links and alternate links that may be used in the event of a faulty link failure. Dijkstra’s algorithm for an all-paths execution has a running complexity time of which can be a burden and a hassle on large networks. The theory presented in the successive sections seeks to employ Dijkstra’s all-paths algorithm only upon network initialization or when critical dependent links have been severed. This severing of a dependent link forces the network to run Dijkstra’s all-path algorithm to locate additional alternate links for the network. This strategy does take into account an exhaustion of previous located alternate paths and is therefore used as a last resort or survival strategy.

Next, once all of the links have been located and recorded, a second and faster algorithm is used to create a minimum spanning mesh network. An adequate approach is to use a familiar and modified minimum spanning tree (MST) algorithm such as Prim’s or Kruskal’s. Both are greedy algorithms and ensure that the minimum spanning tree obtained is the most optimal choice. For our application, we chose to run Prim’s as it was more intuitive for our resolution of the self-healing mesh network. Prim’s running complexity time of , where V are the number of vertices and E are the number of edges, is then used to create and maintain the MST of the network.

The self-healing mesh network takes advantage of both functional portions of the network. Namely, locate all available paths in the network and locate the minimum distance between two nodes intercommunicating with each other within a network. For this reason, the self-healing mesh network strategy is to initially have a single master node in the network run an all-paths algorithm to locate every link available in the network and store a record for each destination such as the table shown in Figure 4.

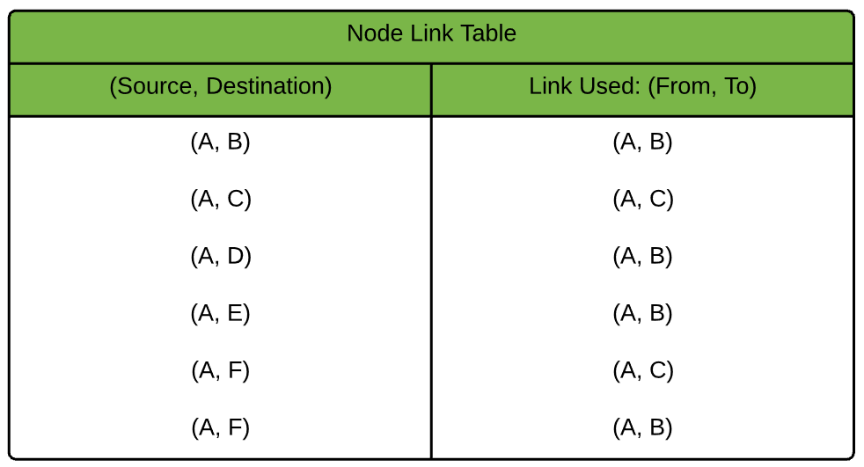


Figure 4: Node Link Table used at each node to help in locating alternate routes to different nodes in the network.

Notice that node A has two alternate paths to get to node F. Namely, it allows node A to use the link from A to B or the link from A to C to get to node F. This is the critical feature that provides the robustness of the self-healing network. Moreover, since each node populates its own Node Link Table, each node knows how to locate any node in the network and which link or links to use in order to send and receive from a source to a destination. The master node is only used to locate every link and once executed, the master node and remaining nodes run a minimum spanning tree algorithm in the network to maintain a shortest paths network but may use the Link Table to seek alternate paths in the event of a link failure.

**Implementation:**

The implementation of the self-healing mesh network relies on the initialization of the network itself. The steps set forth can be performed on any node in the network and uses concepts from Dynamic Source Routing (DSR) protocol but changes the way in which the nodes store the routing information. Additionally, the broadcasting messages for the route request mimic DHCP requests which helps the overall network find every possible link between all of the nodes. The implementation of the network begins by choosing a master node to run a link-state algorithm such as Dijkstra to begin locating all possible links in the network. The requesting node A sends out a Network Link Locating Request (NLLR) to its immediate neighbors while requesting a route to node N as show in Figure 5. The NLLR is a simple broadcast mimicking DHCP requests to any nearby nodes that are potentially listening.

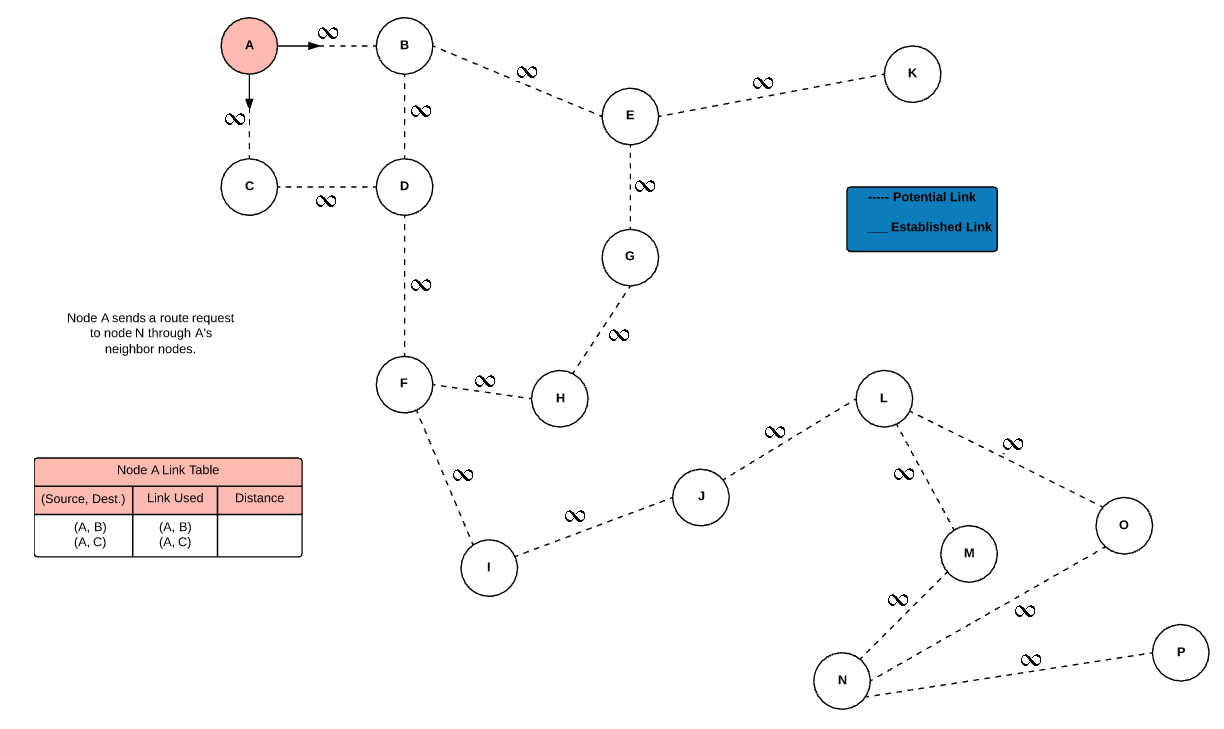


Figure 5: Network initialization begins with a request to immediate neighbors while trying to locate a possible route from source node to destination node.

The source node A at this moment is considered the master node as the network and network links have yet to be initialized. As the requests reach the neighbor nodes, the master node waits for either a route to the requested destination or a distance value to the current node transmitting the distance value back to the requesting node. This is shown in Figure 6. Since nodes B and C are not the destination nodes, they retransmit the route request to their immediate neighbors and send the distance value to the requesting node A. Node A receives the distance values for it’s neighbor nodes and stores the link used and distance to the neighboring node in its Link Table. The Link Table is critical as duplicate source to destination entries are allowed and the table is consulted whenever a link or node fails in the network.

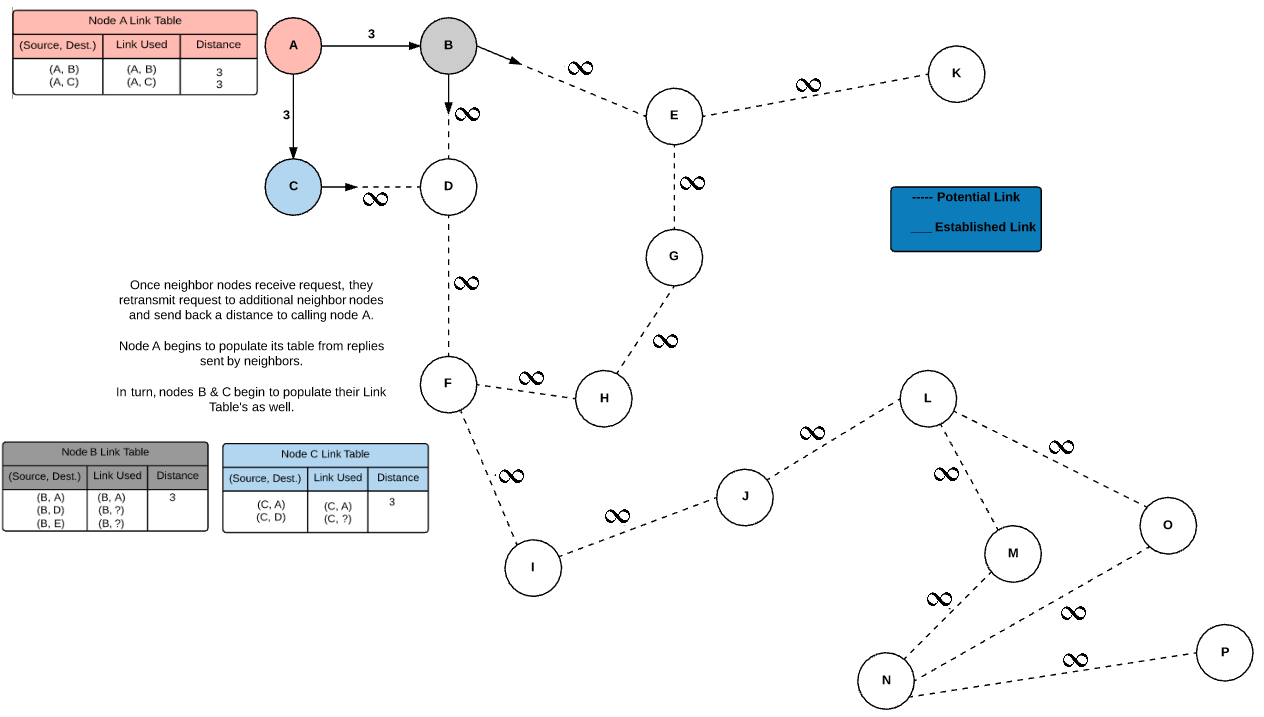


Figure 6: Second stage of the network discovery begins. Both nodes B & C begin to broadcast node A's request to obtain a route to node N. The links discovered thus far are populated in Node A's Link Table and both nodes B & C begin to populate their own Link Table’s.

Consequently, node A’s Link Table is updated to reflect the distance values sent back to it and are stored to determine a route to a future destination node. Moreover, the Link Table’s for node B & C begin to populate as well and are used to send either distance values back to node A or the actual route located. The second stage of the network initialization begins with both node B & C broadcasting node A’s request to their own immediate neighbors in the same fashion. Figure 7 shows how the third stage would look like once nodes B & C have received pertinent network information from their neighbors. Moreover, Figure 7 shows that duplicate entries from source node to destination node are allowed as long as the link used is not the same. This is a key feature in determining an alternate route from source node to destination node in the event of a link failure.

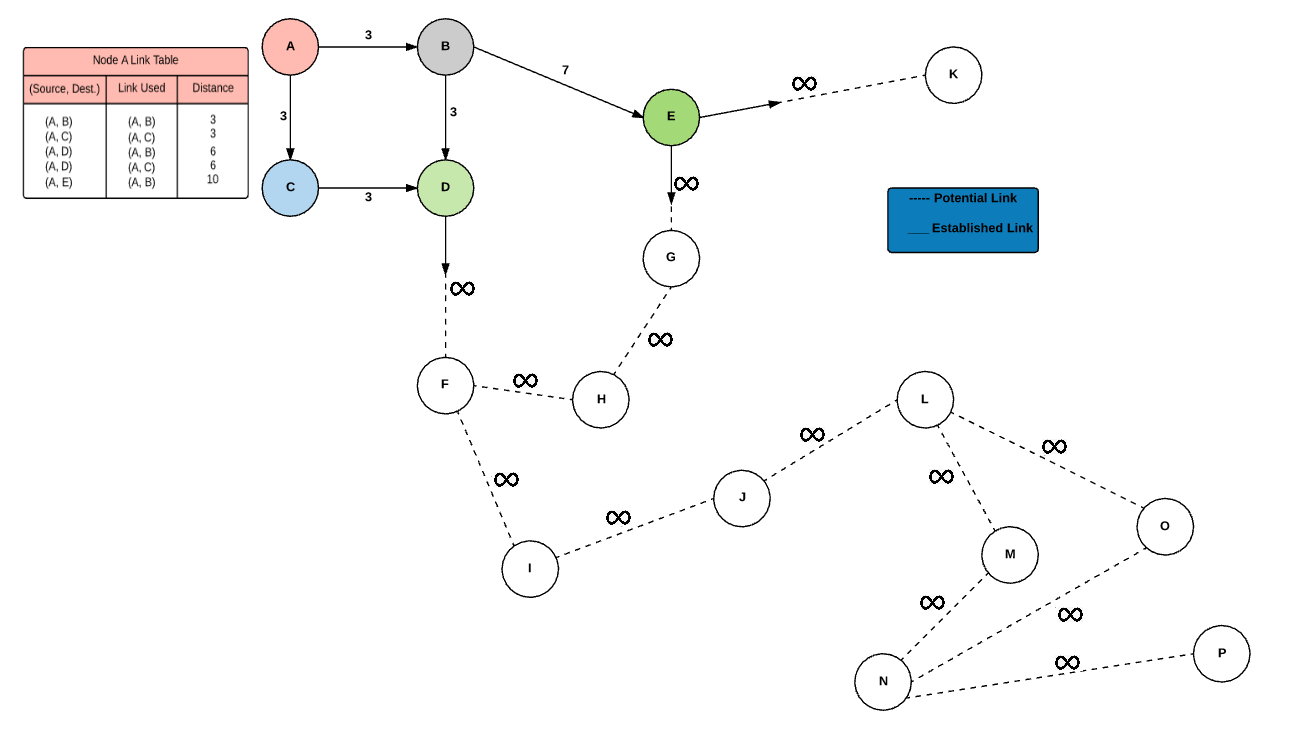


Figure 7: Third stage in the network initialization. Nodes B & C continue the request and send back any information received from neighbor nodes to requesting node A.

Proceeding in similar fashion, the fourth stage continues the trend of locating each possible link and sends the distance information back to the requesting nodes. Node A receives the distance value that is summed up and stores the total distance to the called node in its Link Table. Figures 8 and 9 show the successive request calls to neighbor nodes and the completed network, all its links, and a completed Link Table for node A. The completed table shown in Figure 9 now contains all possible paths from source node A to destination node N.

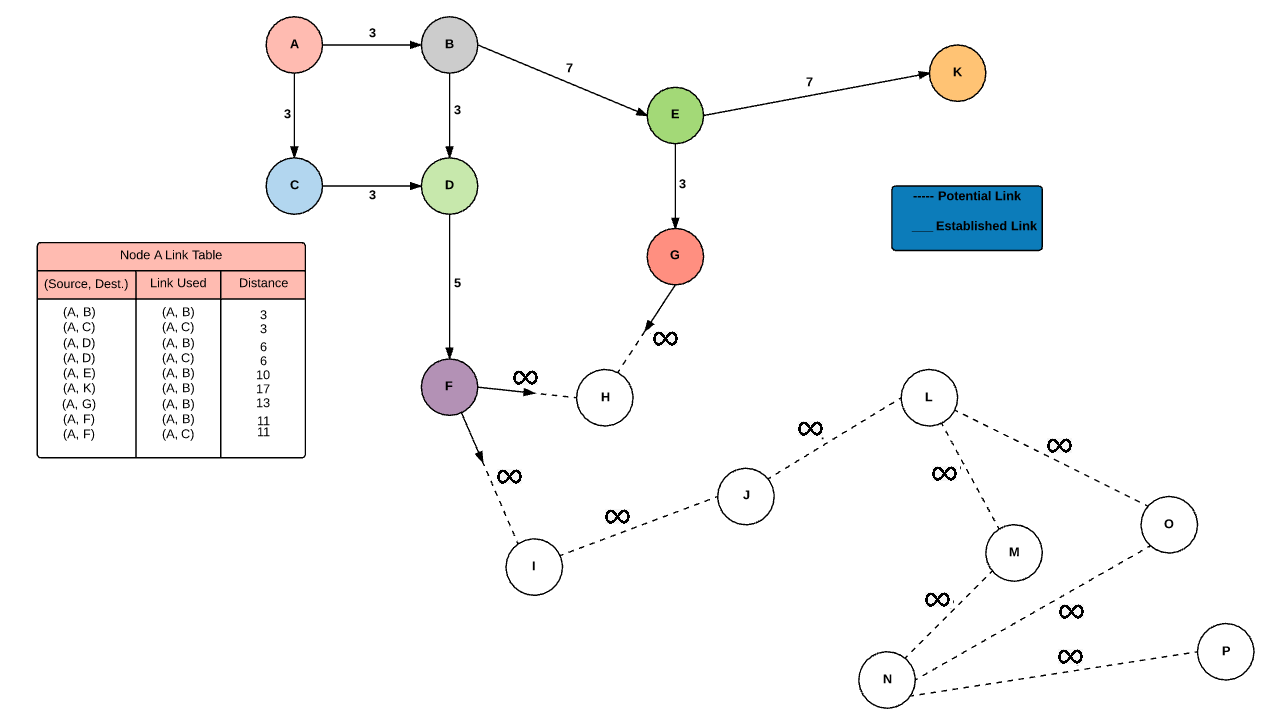


Figure 8: Fourth stage of network initialization. Nodes F & G continue to broadcast network neighbors trying to locate the destination node. If not found, the distance values to nodes reached are sent back to requesting node A and node A's Link Table is updated.

Remember that all nodes discovered so far also have their own set of Link Tables being simultaneously updated as new link information is returned to broadcasting nodes. Figure 9 shows a completed table for node A but all other nodes in the network will contain similar Link Tables reflecting their own distances to all other nodes in the network that have been contacted so far.

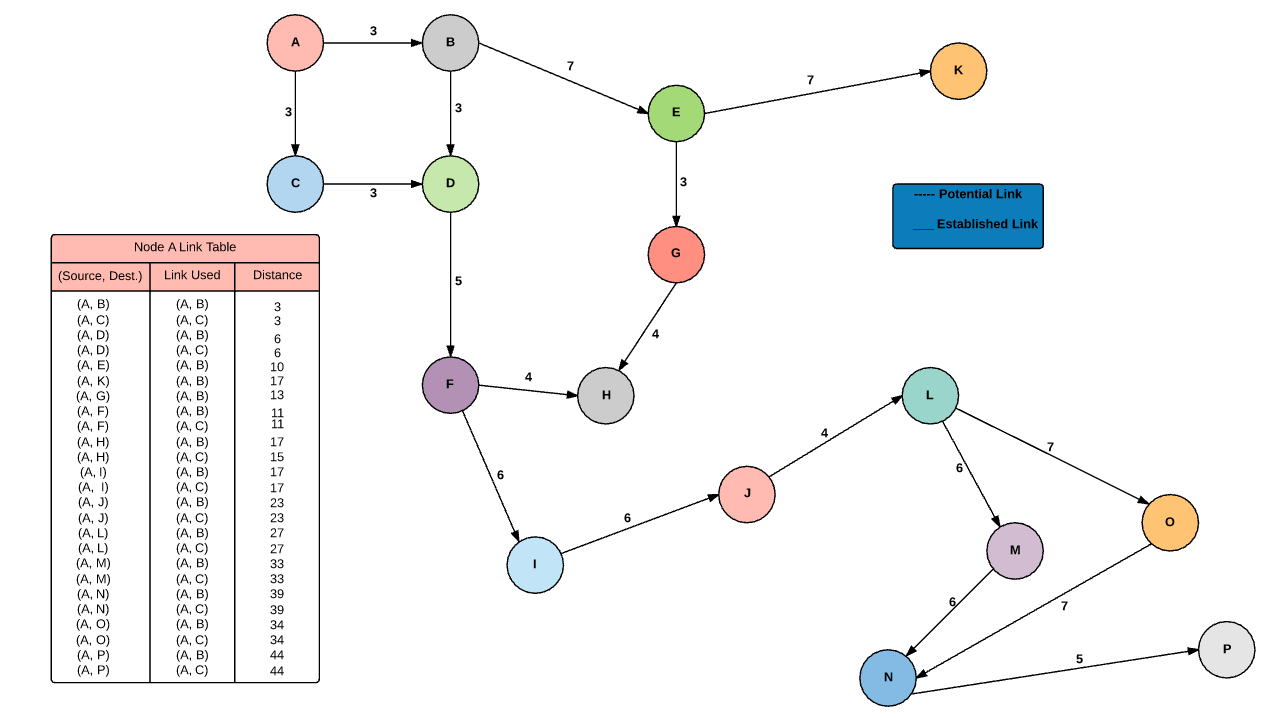


Figure 9: Fifth stage in network initialization. All links have been located and distance values have been sent back to requesting node A. Node A's Link Table is updated to reflect the distances found.

Once the network has been initialized, the next step is to locate the minimum spanning tree (MST) for each node. Assume that a network confirmation bit is sent to all nodes in the network once all links and nodes have been located. As soon as the network confirmation bit is received, each node will use its updated Link Table to find the minimum spanning tree to all other nodes in the network but will not delete any entry in the table as duplicate entries from source to destination are used in locating an alternate route in the event of a link failure. Figure 10 shows the minimum spanning tree for node A.

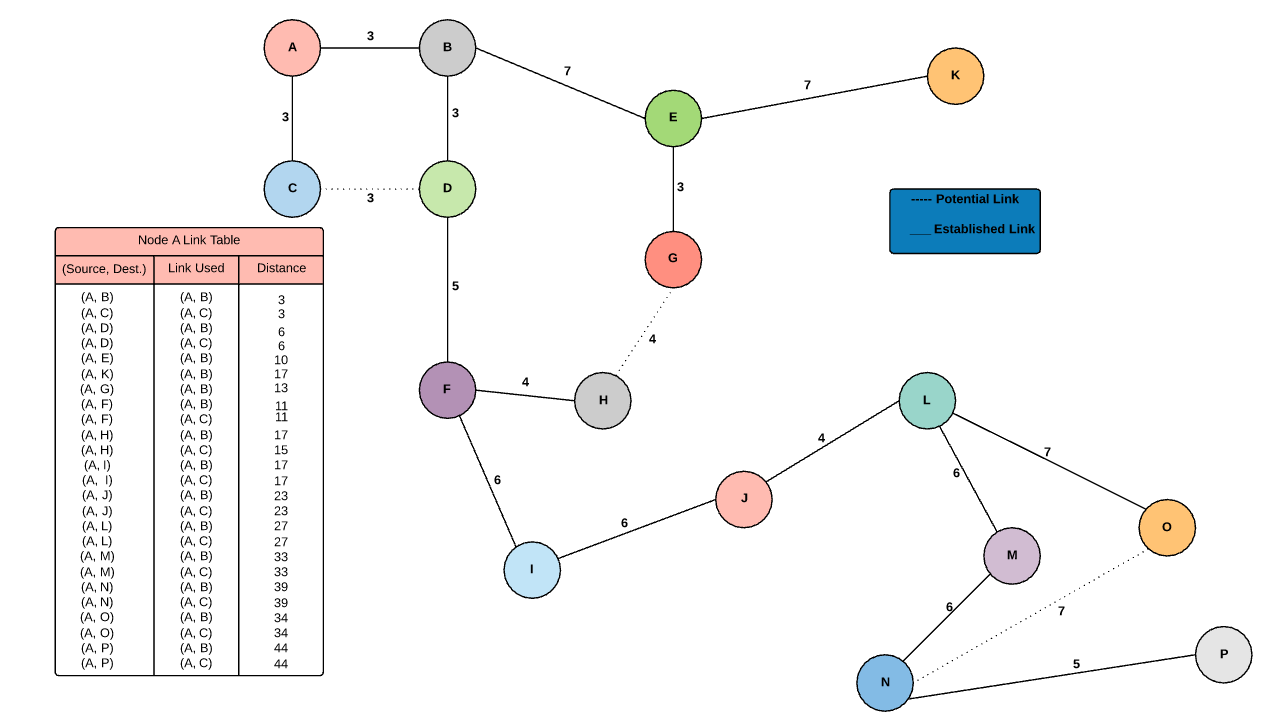


Figure 10: Minimum spanning tree for node A. This MST is used in traversing the network from source to destination. The tree is continually updated in the event of a link failure.

The emphasis of the self-healing mesh network is in the robustness to handle link failures as they dynamically occur. Node A has two links that it can use to get to node D. Namely, and can be used to reach node D and any other node that node D is linked to such as node I or node J. Figure 11 shows how this is theoretically accomplished. As node A tries to send information to node D, it receives an error message from node B stating that the message was not received by node D. Consequently, node A consults it’s Link Table and is able to locate an alternate route through node C in order to get the information to node D.

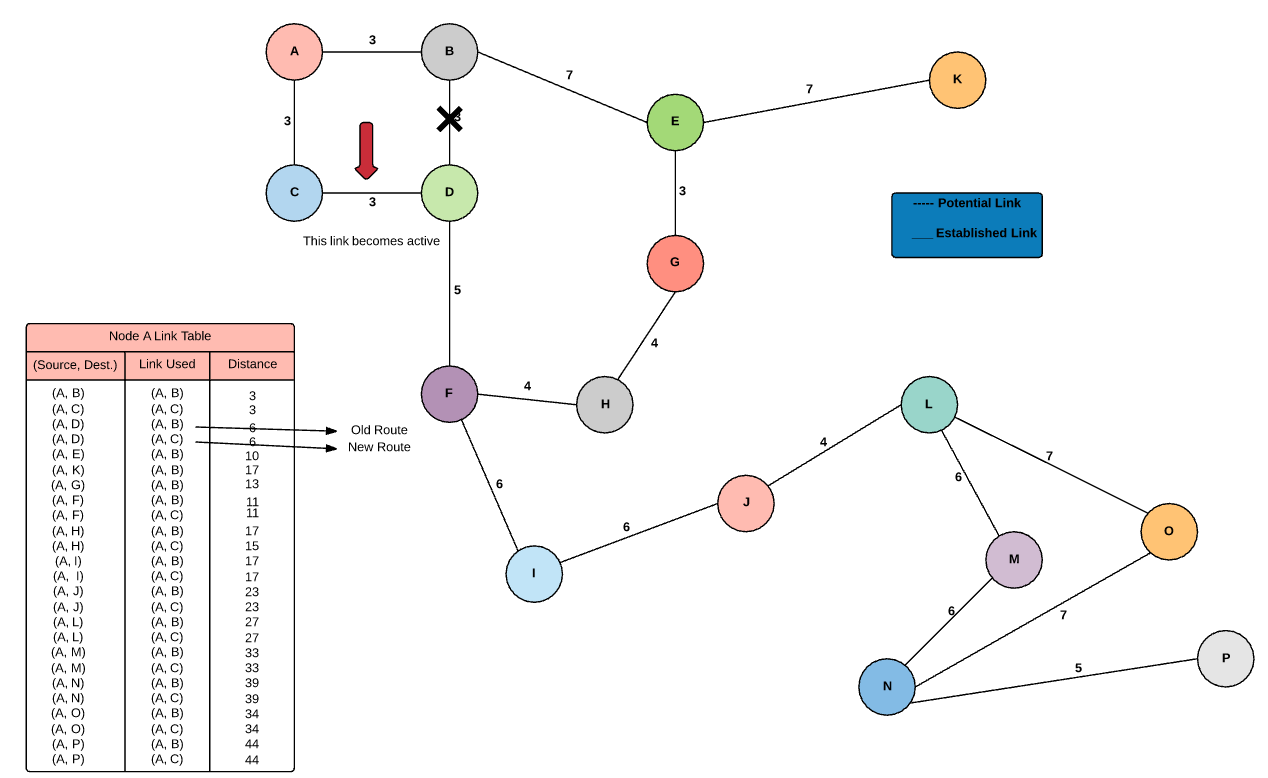


Figure 11: Node A attempts to send information to node D but node B returns a transmit fail message. Node A then uses it's Link Table to locate an alternate route to node D via node C and the MST is updated if successful.

If node A’s request to transmit to node D is satisfied, then node A’s minimum spanning tree is updated. Once the failed link is repaired, then a request to rebuild any node’s MST can be sent to update the shortest path for each node but the Link Table for each node always remains the same unless deleted manually by the network administrator. Figures 12 and 13 shows similarly what occurs when more links fails throughout the network. Each node that encounters a transmission failure will update their MST to accommodate the new route until the primary link has been restored or manually deleted.

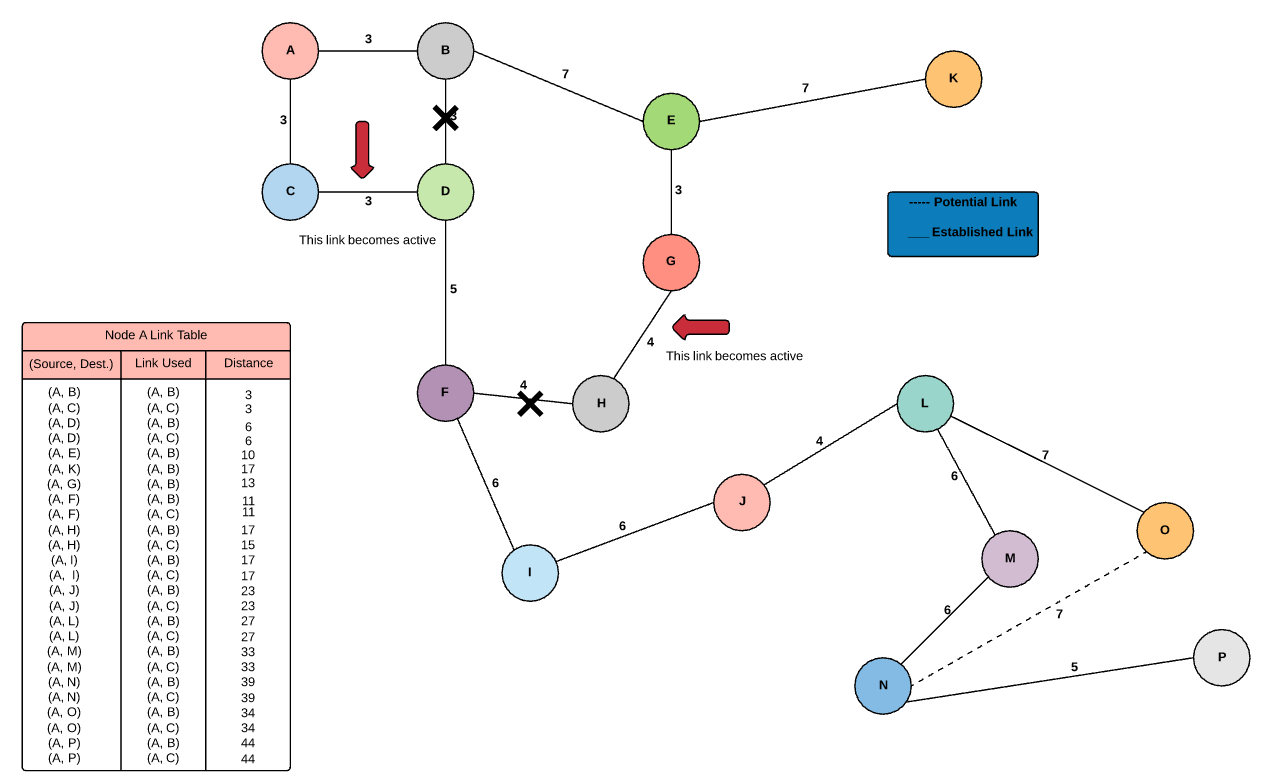


Figure 12: A second link in the network fails. Therefore, affected nodes use their Link Tables to locate alternate routes and update their MST's to reflect the newly obtained route.

Keeping in mind that each node has its own set of Link Table’s, the interrupted node is able to locate an alternate route if an alternate route does indeed exist. Each time a link fails, the table is consulted by each node affected and a new MST is created if successful transmissions to the destination node are received.

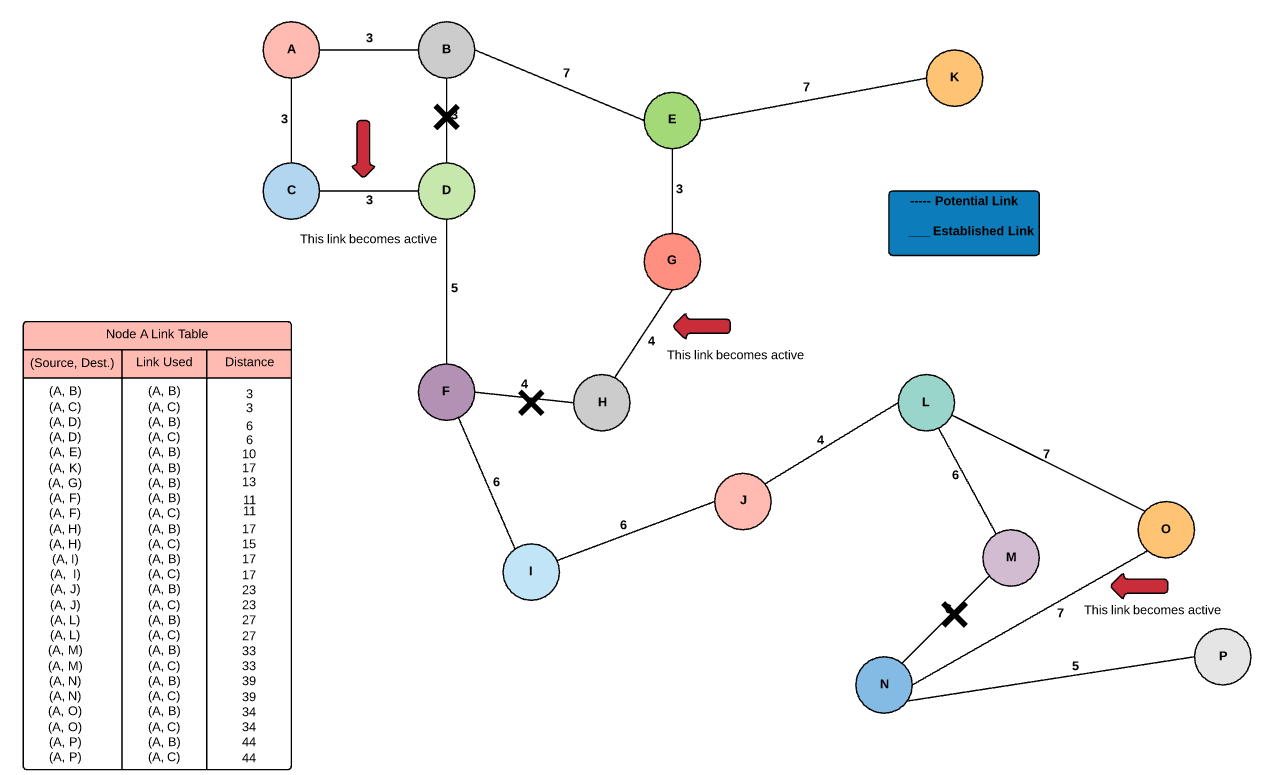


Figure 13: A third link fails in the network. Affected nodes consult their Link Table's to redirect incoming transmissions.

The main features and aspects of the self-healing mesh network are showcased in Figure 13. Although the newly established routes are longer than before, the transmission of information from a source node to destination node still occurs ensuring complete network traversal between any set of nodes.

**Implementation Results:**

The initial state of the self-healing mesh network is shown in Figure 14 below. Figure 14 was made to mimic the same idiosyncrasies as the examples previously mentioned including the depicted nodes used. The direct initialization of the mesh network begins as described above and the links to every node available are stored separately in their respective Link Tables. The test to check if a node was active and receiving information was performed after the initialization ensuring safe packet traversal between the various nodes.

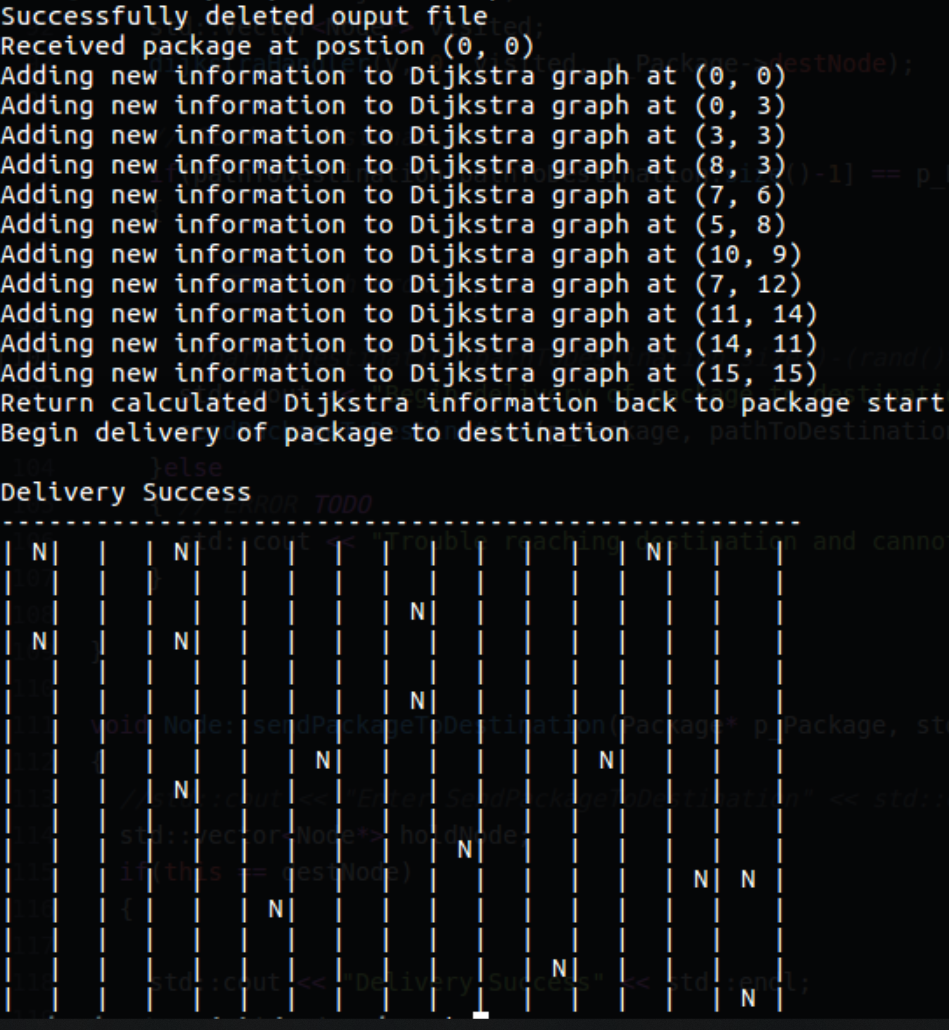


Figure 14: Self-healing mesh network initialization. The initialization is performed using a link-state algorithm such as Dijkstra’s all-paths algorithm to locate all of the potential links in the mesh network.

Although it was tough to replicate the links as previously shown, the test was successful in delivering packet information to the various nodes scattered throughout the network. Once the initialization of the network was complete, the next phase was set to begin building the MST for the network using the different links discovered throughout the network. The MST for the network shown in Figure 14 is built and delivery attempts are made after a single link fails. Namely, link (3, 3) fails after a no response timeout and the self-healing network begins to attempt a repair on the network transmission that previously failed. This test was conducted to ensure that the affected node was able to consult its Link Table to seek out an alternate route. Looking at its Link Table, the affected node is able to traverse the network links and locate an additional path located at (2, 8). Figure 15 shows the final outcome of the network repair. Once the alternate path has been located, an additional attempt to deliver the payload to the source destination located at (15, 15) is made. If a confirmation is returned by the destination node, the link repair is conserved to be successful and the newly formed MST is saved to the network to use until the offending link or node has been repaired.

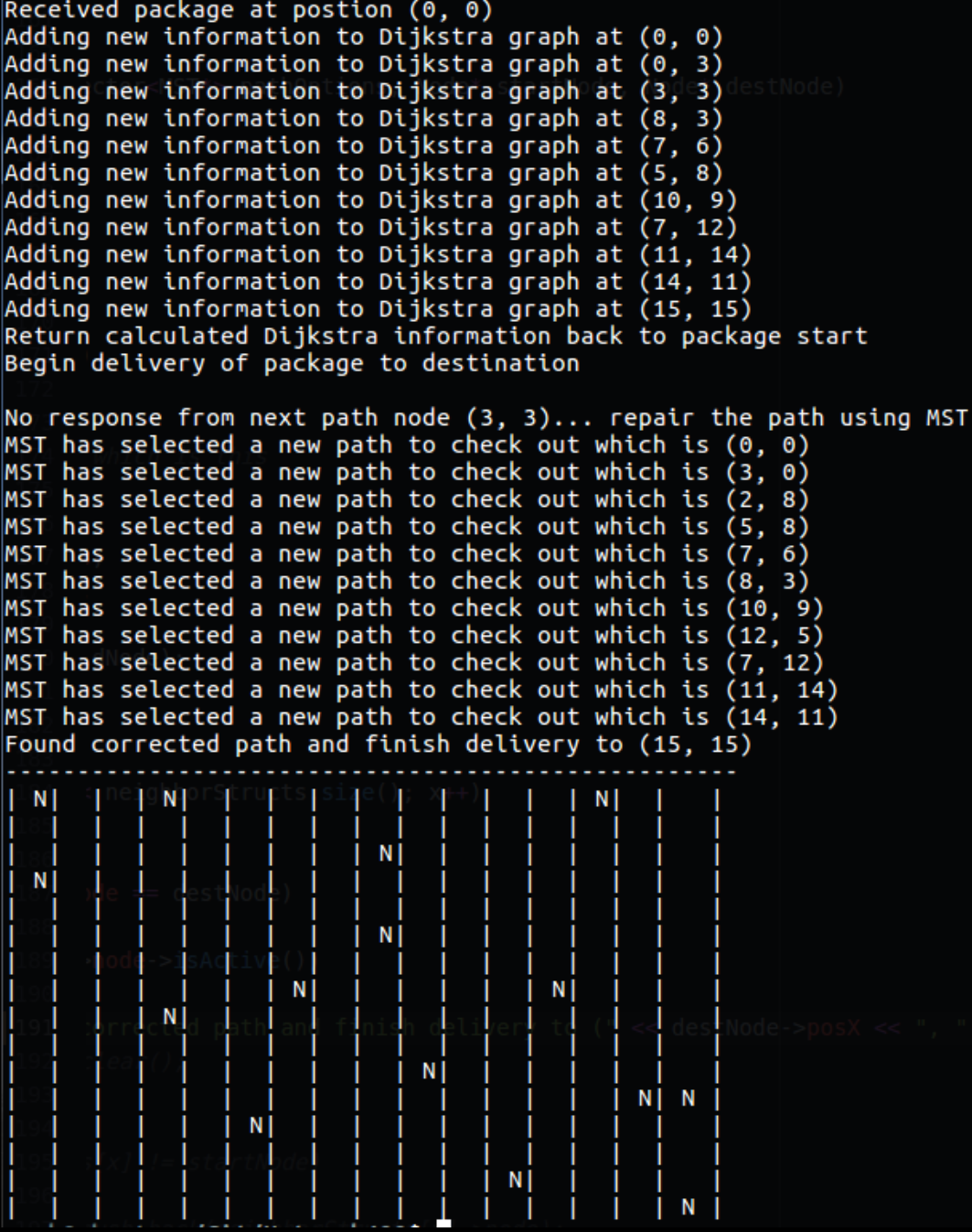


Figure 15: Network traversal fails at (3, 3) so an alternate route is calculated using the Link Table at the affected node. Success is determined if the newly calculated route from source to destination is confirmed. If successful, the newly created MST is saved for network traversal until the offending node or link has been repaired.

The safe delivery of the payload to the destination node proved successful in the tests ran. The nest phase of the self-healing network was to determine if the network would fail when a dependent link of node has indeed failed. We decided ultimately to not fuse or merge the links if a dependent link failed and not to repair an offending node if it were to also fail. The tests were performed to showcase the vulnerabilities of the self-healing network given the exercise network depicted above. The preliminary test started by manually deleting a dependent link that effectively severs the transmission between the source node and the destination node. The test results are shown in Figure 16.

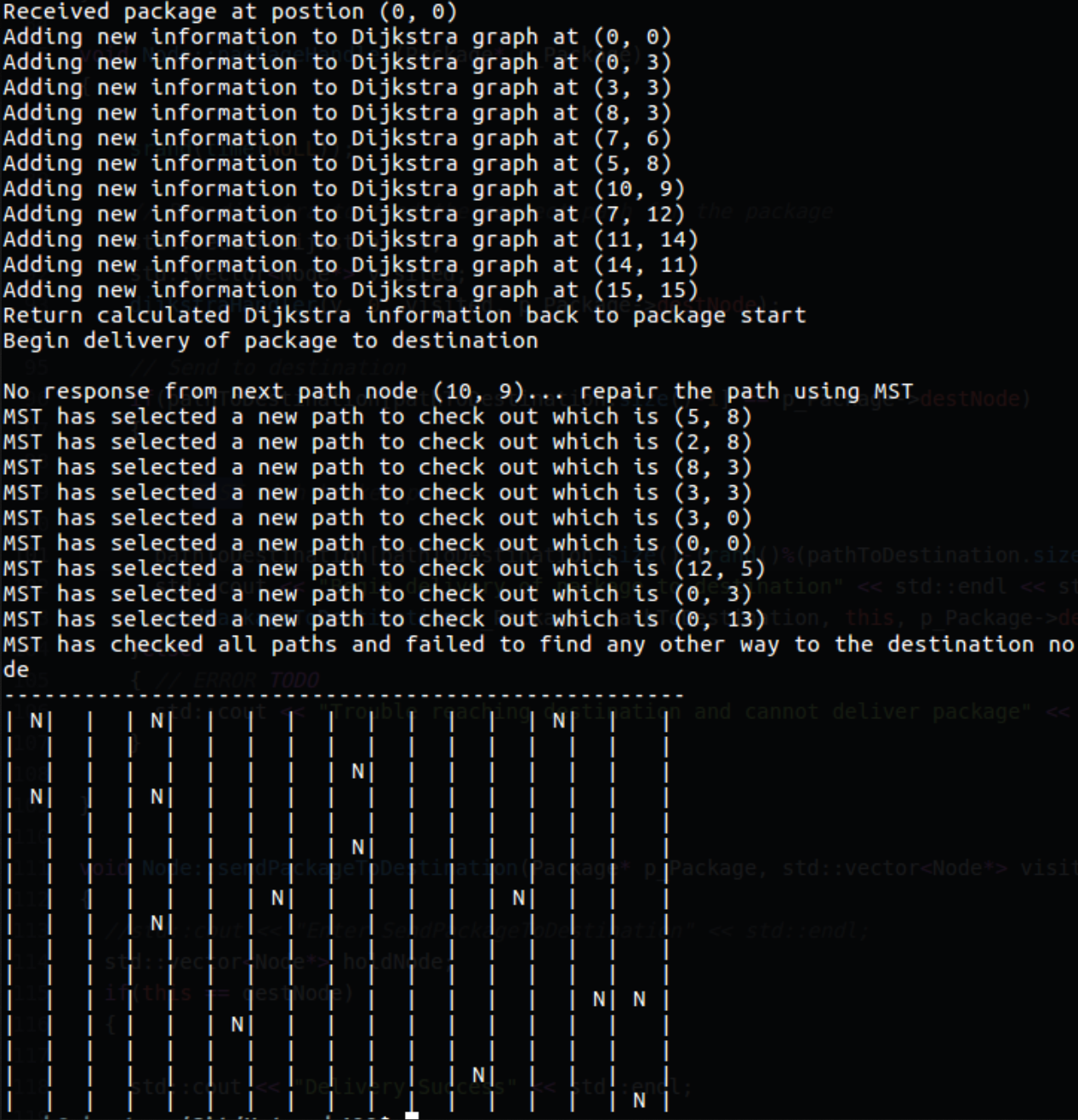


Figure 16: Dependent network link fails, and the affected nodes Link Table fails to find an alternate route to the destination node. The failure is recorded with the failed link and an error message is shown to the networks users.

The offending link presiding to node (10, 9) is found to have failed as a timeout occurs. The affected node considers possible alternate routes stored in its Link Table and is unable to find an alternate route which generates an error message. At this point, outside intervention is needed and the offending links can be fused together to complete the connection, or the node must be repaired. A proposed solution is mentioned under the network issues section below. Lastly, the error ping response is shown in Figure 17 indicating the error.

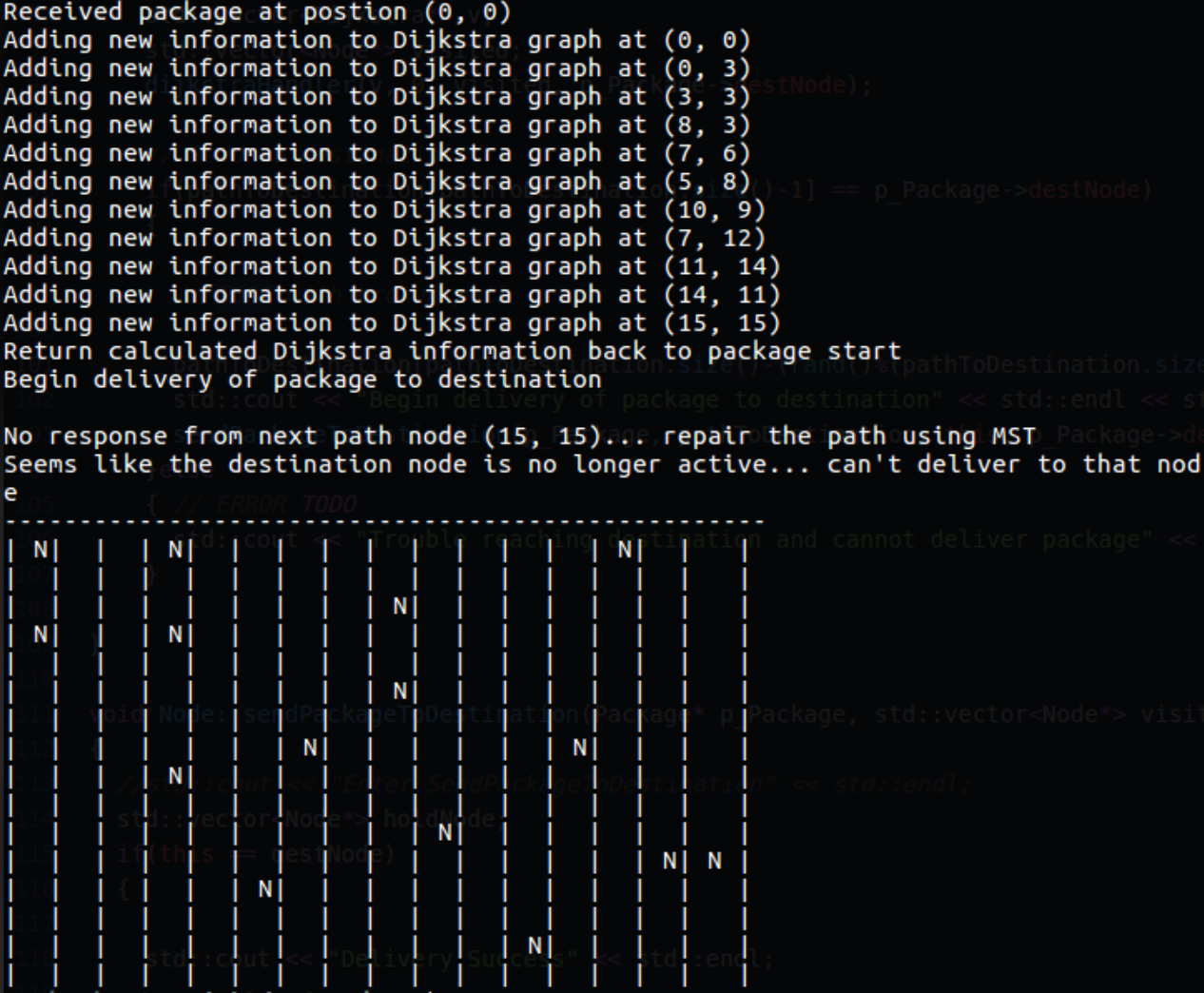


Figure 17: An error ping is sent back, and the message is displayed to the networks users.

The examples mentioned above offer only a small sample of the capabilities and vulnerabilities of the self-healing network. Suggestions to improve the networks robustness include having at least two links available on nodes that are considered to be network dependent. This approach would ensure at least a single alternate to exist between the various nodes scattered throughout the network, but also adds the complexity and real-life costs as each link requires extra funding to create and maintain.

**Self-Healing Mesh Network Issues:**

The issues mentioned here are mostly the worst-case scenarios as dependent links or nodes are the root issue. The first major obstacle that can render some or all of the network useless is if a dependent link ever fails. Consider if link were to fail on Figure 13 after the first three failed links scenario has occurred. In this worse case failure scenario, half of the network is no longer reachable until the offending link is repaired. This critical link might need to have a counterpart to ensure traversal throughout the network. Consider a remedy to the potential crippling issue by implementing a new link from or a new link from . This would preserve the robustness of the network in the event of a link failure but adds extra overhead to the overall network including the Link Tables that must be maintained at every node. Each new link added ensures a more robust and tolerant mesh network but gets closer to a full mesh topology. The full mesh topology is being avoided here assuming the network can exist without every link being mapped to save monetary and running time costs.

Another potential issue is the permanent deletion of a dependent node. For example, if node J on Figure 18 was permanently deleted, then its corresponding links are considered dead and the right-half of the network is no longer reachable. In this worst-case scenario we have a few options left at our disposal. The first remedy is obvious and repairing the node to its normal operating state or creating a simple routing mechanism would fix the issue. The second is to combine the links that lead to node J and update each Link Table in the network. This approach is shown in Figure 18.

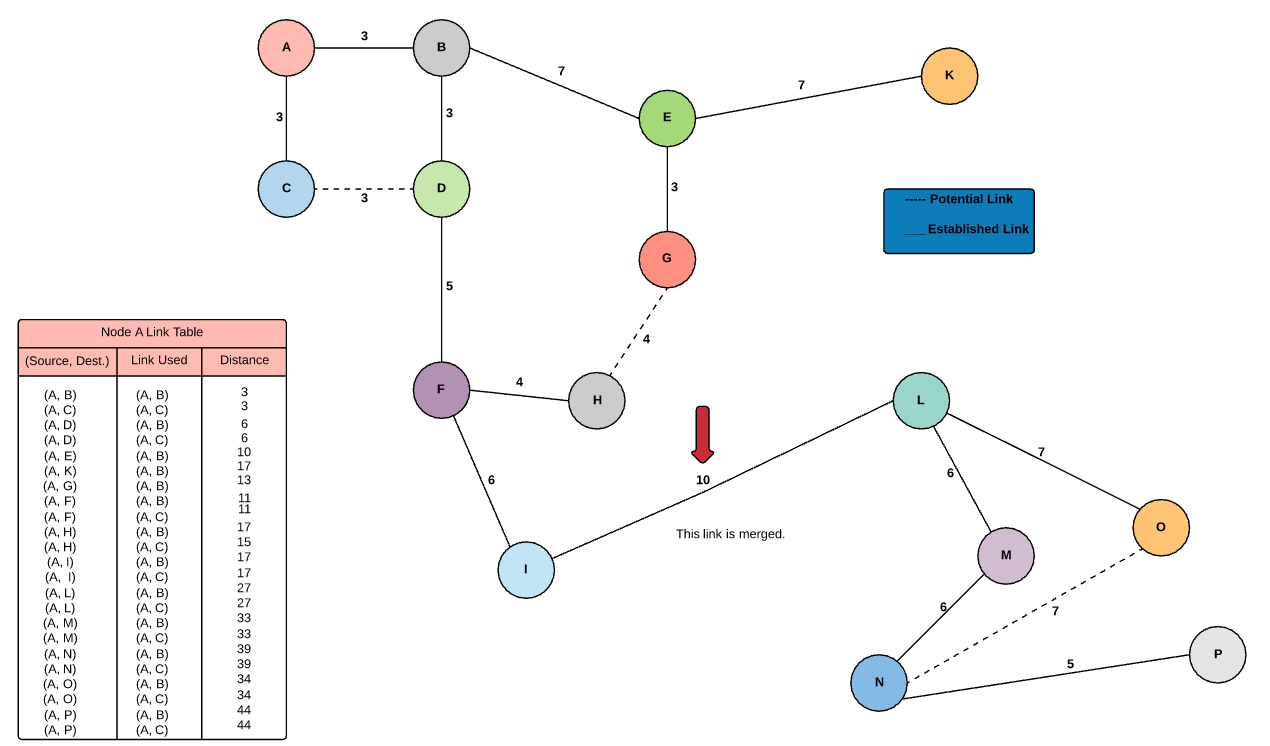


Figure 18: The link merger resolution approach is shown. Links are merged together to keep the network operational and the Link Table's for each affected node is updated.

The link merger approach tends to be the easier solution if the offending node is permanently deleted from the overall network. Unfortunately, an issue with the link merger method also emerges. Since the link is restored and updated with a new value, all of the nodes in the network must now delete the node permanently from their Link Table’s, update the Link Table, and redo the MST for each node. This increases the time to recover for the network but restores the network’s robustness. Some systems may be able to tolerate the network setup running time but networks with continuous traffic are affected severely.

**Conclusion:**

The self-healing mesh network theoretically is a great addition to a network that must be kept constantly alive due to its users. Issues are abounding on any network scheme but tolerable to certain network applications. The self-healing network approach presented here is based on current algorithms used in major communication networks but can be altered to fit any business model. The key aspects include a safe, reliable, and fault tolerant network that can be used with cost cutting techniques and measures for nodes and links used throughout the network. Future work on the network involve decreasing initialization time for the entire communication network as the initialization of the system is the bottleneck.

Cited Sources

1. S. Büttrich, “Mesh Networks.” 07-Nov-2005.
2. M. Faruqui, A. Tripathi, “Fault tolerant routing in mesh topologies,” IIT Kharagpur, Kharagpur, West Bengal, India. [Online]. Available: http://cse.iitkgp.ac.in/~anshumant/reports/fts.pdf. [Accessed: 29-Nov-2017].
3. J. Henry, “802.11 mesh networking,” CWNP, Durham, NC, USA, 2011. Available: https://www.cwnp.com/uploads/802-11s\_mesh\_networking\_v1-0.pdf. [Accessed: 29-Nov-2017].
4. T. Li, “How do mesh networks work?,” *Quora*, 03-Jun-2015. [Online]. Available: https://www.quora.com/How-do-mesh-networks-work. [Accessed: 29-Nov-2017].
5. “Mesh Topology,” *The Network Encyclopedia*. [Online]. Available: http://www.thenetworkencyclopedia.com/entry/mesh-topology/. [Accessed: 29-Nov-2017].
6. “Our guide to network topologies,” *Help with PCs*. [Online]. Available: https://www.helpwithpcs.com/hardware/network-topologies.php. [Accessed: 29-Nov-2017].
7. “What's a mesh network?,” *eero Help Center*. [Online]. Available: https://support.eero.com/hc/en-us/articles/207646676-What-s-a-mesh-network-. [Accessed: 29-Nov-2017].