Integrating Network Discovery with Network Monitoring: The NSFNET Method

William B. Norton <wbn@merit.edu>

Abstract

Network discovery algorithms are often separated from network state determination algorithms. During the management and operation of the NSFNET, we have found significant benefits from combining network discovery and state determination.

This paper describes the network management discovery algorithm used to determine the state and topology of the NSFNET. This algorithm makes the important distinction between nodes and links being up, down, and not reachable. In doing so, it automatically supports the notion of dependency. This simple algorithm also deals well with the condition of incomplete network information, which often exists in a busy or congested network.

I. Initial Discovery

Discovery involves collecting network topology information from the network. The basic requirements for discovery to work are network connectivity, a network topology and state store, and the means to interrogate the network. We'll discuss the topology and state store first.

The node and link state store is required to hold the state and topology information. Associated with each node is a state variable, indicating the probable state of the node. There are only three states a node can take:

UP: indicates that the node responded completely to the last poll.
NR (Not Reachable): indicates that the node has not responded to the last poll, and no node has reported adjacency to it during the last poll.
BUSY: indicates that an adjacent node claims the node is responsive, but the node itself has not responded completely to the last poll

In these node state definitions there is no DOWN state. This is because there is no way to determine that a node is, in fact, down. Connectivity may be preventing the node from being responsive. Further, any level of backup connectivity may also fail. While one can make a probabilistic determination that the node is likely to be down, we have found this not to be useful.

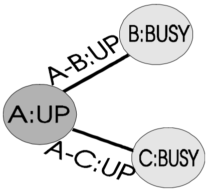
Links, unable to respond for themselves, can take on the following states as dictated by their neighbors:
UP: indicates that a node at either end or both ends of the link claimed the link is usable during the last poll.
DOWN: indicates that either end of link claim the link is unusable during the last poll.
NR: indicates that no node reported the existence of this link during the last poll.

The discovery algorithm requires a "seed", that is, at least a single query able node in the store. In this case (see Fig. 1), the store consists of a single node A which, since we haven't heard from yet, is assumed to be in the NR (Not Reachable) state. Note that at this time, no links are known, so the link store is empty.

Node	State	Link	State
A	NR		

Figure 1. Seeding the Discovery Algorithm

The first poll consists of adjacency queries of the only node in the store . Node A reports adjacency to node B and C and claims that both links are in the "UP" state (see Fig. 2). The fact that A responded is necessary and sufficient to declare that node A is in the "UP" state. Further, A claims that both links "A-B" and "A-C" are "UP" which is necessary and sufficient to declare that links "A-B" and "A-C" are in the "UP" state. This is based on a major assumption described next.



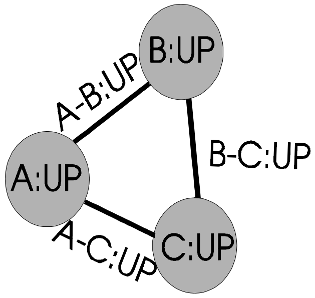
Node	State	Link	State
A	UP	A-B	UP
B	BUSY	A-C	UP
C	BUSY		

Figure 2. Topology after one poll

This discovery algorithm expects that each node can tell with some certainty the state of its neighbor. In the NSFNET, the IS-IS link state protocol is used to determine the availability of the links. The method used to determine the state of the link is to send HELO packets from peer to peer periodically. If a HELO is received, the node declares the link to be UP. One can therefore infer that if the node claims that the link to neighbor is UP, then the neighbor (peer) must also be UP. Conversely, if a HELO packet is not heard from a neighbor for some period of time, the link is marked as DOWN and is no longer used.

After this first poll, we find that node A knows it has a functional link to nodes B and C so these links are added to the store. Further, since we know these links are functioning, we can infer that B and C are most probably UP, but we haven't heard directly from them. Since this is the definition of the "BUSY" state, both of these nodes are added to the store in the "BUSY" state. Note that in one poll (of only one node) we discovered three nodes and two links.

The algorithm now polls the three nodes A, B, and C for adjacency information, and the store is updated as in Fig. 3. Since all three nodes respond in this case, we know all three nodes are up and are marked in the "UP" state. In this scenario, nodes B and C both declared link B-C as being UP, so this link is added to the store.



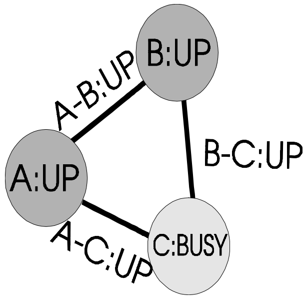
Node	State	Link	State
A	UP	A-B	UP
B	UP	A-C	UP
C	UP	B-C	UP

Figure 3. Topology after two polls

II. Incomplete Information

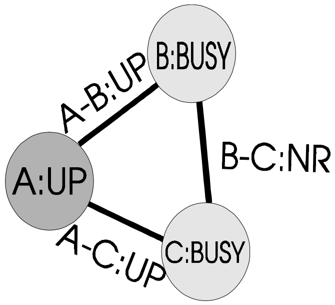
In certain situations, (network instability, node resource starvation, link congestion, etc.), some nodes may be non-responsive. As noted earlier, this algorithm deals with this situation through the BUSY state. As long as either end of the link declares the link to be usable (and neither claim it is unusable), the link remains in the UP state. Thus, the failure of any one node to respond will have minimal impact on the operation of the algorithm.

In the next case (see fig. 4), an unresponsive node C does not cause any degradation in network state information because of the ability to infer the neighbors state. To take this one step further, assume that (see fig. 5) both B and C do not respond to a poll. Since A reports functioning links to both B and C, but neither B nor C responded, B and C are both in the BUSY state. Since neither B nor C reported link B-C, this link is determined to be in the NR (not reachable), or "unknown" state. Thus, no false alerts are reported, and the store accurately represents the state of the network.



Node	State	Link	State
A	UP	A-B	UP
B	UP	A-C	UP
C	BUSY	B-C	UP

Figure 4. Incomplete information: The BUSY State



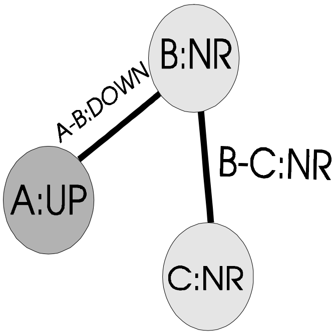
Node	State	Link	State
A	UP	A-B	UP
B	BUSY	A-C	UP
C	BUSY	B-C	NR

Figure 5. Incomplete information: The NR Link State

In fact, during the management and operation of the NSFNET we have found that incomplete information is returned more often then one might expect. Routing updates occasionally require great node resources. Phone company lines experience intermittent glitches that may be corrupting traffic. Using certain vendor's equipment, these events may cause the network management system to erroneously alert the Network Operations Center (NOC) and declare the node to be DOWN or unreachable. This algorithm is relatively tolerant of these faults, since it bases the determination of node and link state on both the reachability information and the neighbors notion of the network state.

III. Dependencies

Dependency, in the context of network management, means the ability to differentiate between network outages and the side-effects of those outages. Obviously, the notion of dependencies is critical to the effective management and operation of a large internet. Network Management Station (NMS) platforms without this ability will likely create many alerts for a single outage, and provide no indication of the real problem. Consider the following different topology and scenario:



Node	State	Link	State
A	UP	A-B	DOWN
B	NR	B-C	NR
C	NR		

Figure 6. Dependencies: Hiding Side Affects

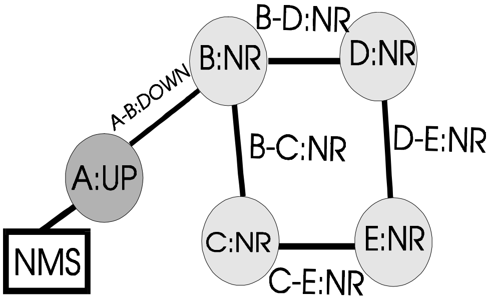
The link between A and B is severed (probably by a back hoe fiber-detector!), and as a side effect, both B and C are isolated. Naturally B and C will not respond to queries, but A will report the link to B as DOWN, and the rest of the nodes and links as NR (Not Reachable). Thus, the algorithm correctly identifies the real and direct cause of the outage (A-B:DOWN) distinctly from the side effects of the outage (B:NR, B-C:NR, C:NR)

The significance of the benefits of this algorithm becomes more apparent when one considers isolation faults in larger environments. The more unreachable nodes that exist on the "other" side if the partition, the more the NOC needs to quickly repair the cause of the isolation. Most currently available NMS platforms would present the cause of the fault along with the side effects of the fault to the operator, failing to distinguish between the two types of information. Network management software is unlike normal application software in that network management software must work best when the network is at its worst. Our experience has been that this algorithm has successfully met that criteria and effectively focuses the network operators attention on the closest cause of the network fault.

Note that if there had been a prior outage beyond B, the store does not save that previous outage state information, but does save the topology information. This is consistent with the philosophy of maintaining "correct" information in the store. The outage in C may or may not still exist, but the true state is unknown until the node or the neighbor respond.

IV. Multiple Monitor Points

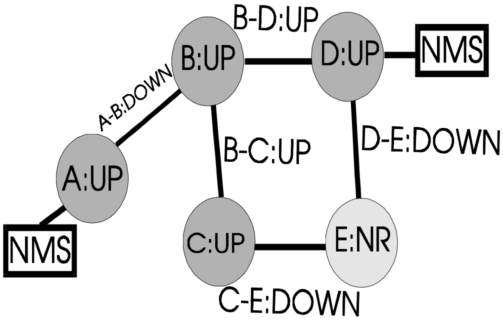
So far we have been assuming that monitoring is done through node A. Thus, dependencies are determined based upon the availability of information along paths beginning at A. The effect of this strategy is that during network partitions, a whole portion of the network will be in the NR state, and only a portion of the network will be able to be monitored. From this vantage point, we know that A is in the UP state, and the link A-B is in the DOWN state. The rest of the network is not reachable, and therefore in the NR state.



Node	State	Link	State
A	UP	A-B	DOWN
B	NR	B-C	NR
C	NR	B-D	NR
D	NR	C-E	NR
E	NR	D-E	NR

Figure 7. Single Monitoring Point

Assume that a second NMS is installed with out-of-band communication facilities to the first NMS. The two combined views of the network, will provide a more complete picture of the network (see Fig 8.) In this case, what was previously unknown (the state of the network behind the A-B isolation) is now known. Thus, we now know that node B is UP, so one problem must be in the link between A and B. Further, C and D are both UP, and we also know that the node E is isolated behind links D-E and C-E. The significant advantage of this distributed management approach is that the simplicity of the algorithm is maintained, and greater information results from additional monitoring points.



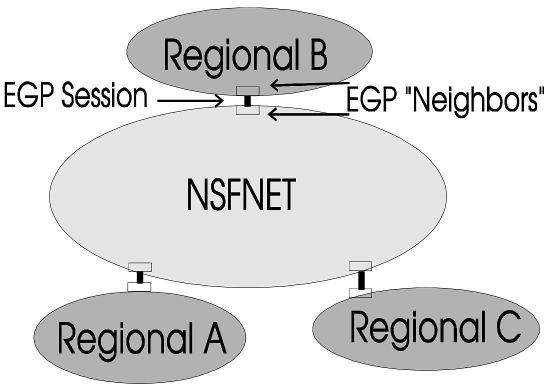
Node	State	Link	State
A	UP	A-B	DOWN
B	UP	B-C	UP
C	UP	B-D	UP
D	UP	C-E	DOWN
E	NR	D-E	DOWN

Figure 8. Second Monitoring Point

The other important benefit of discovering topology as well as state in each poll is that topological changes as well as state changes are recognized immediately after polling. As additional neighbors are attached to the NSFNET, the IS-IS neighbor table increases. Node additions and subtractions are therefore discovered every poll. One problem we've encountered is that nodes may be attached, discovered and monitored prior to being operational! Network operator procedures, contact information, etc. may not be in place at this time, so alerts for non-operational nodes need to be dealt with differently than those for operationl nodes. Similarly, the issue of automated subtractions needs to be addressed (i.e. do you want to automatically remove nodes that are no longer attached; and, if so, does one want the software to do so without operator acknowledgment?, etc.)

V. Algorithm Expanded to Monitor National Regional Network Connectivity

Over the years, regional networks have attached to the NSFNET backbone using a variety of External Gateway Protocols (EGPs); from EGP to the current version of the Border Gateway Protocol, BGP-4. These protocols maintain their sessions by using "keep-alive" packets, much like the HELO packets described earlier under the IS-IS link state protocol. By extending the notion of an attached network to the notion of a "neighbor", this algorithm can be applied to network peer attachments as well. This is accomplished by polling the peer session states.



Node	**State**	**Link**	**State**
NSFNET	UP	NSFNET-A	UP
A	UP	NSFNET-B	UP
B	UP	NSFNET-C	UP

Fig. 9 Regional Networks as Adjacencies

At Merit Network Inc., this algorithm has been applied to monitor the nodes and links within NSFNET proper as well as the regional network attachments to the NSFNET (see Fig. 9) . During the early days of the project, the External Gateway Protocol was the only EGP used, and the MIB-2 defined adequate instrumentation for monitoring the connections between the backbone and the regionals.

However, networking technology changes frequently, and in this case, the routing protocols changed. EGP was replaced with the Border Gateway Protocol (BGP). Unfortunately, too often the instrumentation lags behind that which should be instrumented. This was the case in the NSFNET, where the lack of BGP instrumentation crippled the ability for the algorithm to effectively monitor the regional neighbors.

A “temporary” solution was put into service to circumvent this lack of instrumentation. Since Merit maintains a database mapping regional networks to EGP peers to Autonomous Systems (ASs), and we can determine which ASs are being announced to the backbone, we can infer the state of the attached regional. This, however, negates the intrinsic dependency and incomplete information benefits described in the algorithm, because this reachability information is not retrieved directly from the neighbors. Several years later, BGP is finally being installed on the NSFNET (along with instrumentation!), and this “temporary” fix can finally be dismantled. It is always interesting to see how long temporary kludges live!

VI. Summary

This simple algorithm discovers both the topology and state of a data network, and this paper described its application to the NSFNET. The algorithm uses both declarative states (UP, DOWN) and inferred states(NR, BUSY) to accurately describe the state of the network at any point in time. Dependency and incomplete information are dealt with through the semantics of the state information of the store and the application of the algorithm. The network store is accurate in the context of a single view, but can be made more accurate through the use of multiple network monitoring points.

This algorithm is by no means a network discovery panacea, however. While this algorithm has successfully been applied to the NSFNET for many years and other (non-IS-IS) environments, the requirement that a node MUST be able to infer the state of its neighbors is not met in many environments. For example, on a Local Area Network (LAN), one’s neighbors do not necessarily communicate, and therefore, can not directly infer their neighbors state. Further, as in the case of the NSFNET, even if the routing protocols support discovery, the instrumentation must also exist to perform the algorithm efficiently. For this algorithm to work, the routing algorithms must support the notion of neighbors and neighbor state intrinsically and the appropriate instrumentation must be available.

VII. References

Jeffrey D. Case, Mark S. Fedor, Martin L. Schoffstall, and James R. Davin. A Simple Network Management Protocol. Request for Comments 1157, SNMP Research, May 1990

David L. Mills. Exterior Gateway Protocol Formal Specification. Request for Comments 904, University of Delaware, April 1984

Kirk Lougheed and Yokov Rekhter. Border Gateway Protocol. Request for Comments 1105, Cisco Systems, June 1989

Keith McCloghrie and Marshall T. Rose. Management Information Base for Network Management of TCP/IP-based internets. Request for Comments 1156, Hughes LAN Systems, Inc., March 1991

Marshall T. Rose and Keith McCloghrie. Structure and Identification of Management Information for TCP/IP-based internets. Request for Comments 1155, Performance Systems International, Inc. May 1990

Marshall T. Rose. The Simple Book, second edition.Prentice Hall Series in Innovative Computing. Prentice Hall, Englewood Cliffs, New Jersey, 1993. ISBN 0-13-177254-6

Author Information

William B. Norton is systems research programmer for the Merit Network, Inc., where he has been leading network management activities for the National Science Foundation Network (NSFNET), the big-ten university network (CICNet), the Michigan Regional Network (MichNet) and the University of Michigan (UMNet) since 1988. During this time, he created the Internet Rover package as an integration and network management tool. This tool was released, demonstrated at national networking conferences (InterOp, Net'92), and has been deployed at instituitions around the globe After founding Norton Associates Consulting Inc., Bill has consulted and provided training for various educational, research, and networking companies. He is an active participant in the Internet Engineering Task Force, and can be reached at wbn@merit.edu. Bill holds a B.A. from the SUNY Potsdam.