An Algorithmic Approach to Identifying Link Failures

Mohit Lad
University of California, Los Angeles
U.S.A.

with Akash Nanavati, Lixia Zhang at UCLA and Dan Massey at USC/ISI

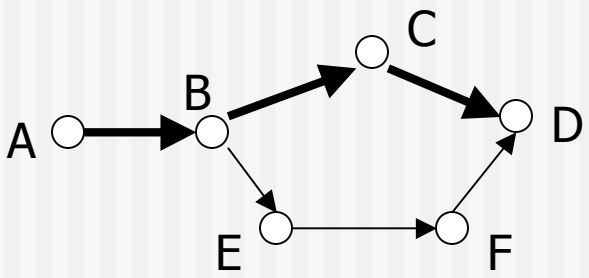
From Nanog Mailing List.

```
Can anyone with a host on 64/8
  try reaching termforsale.net
  or creativesound.ca ?
```

- *From:* Damian Gerow• *Date:* Thu Oct 31 10:09:13 2002

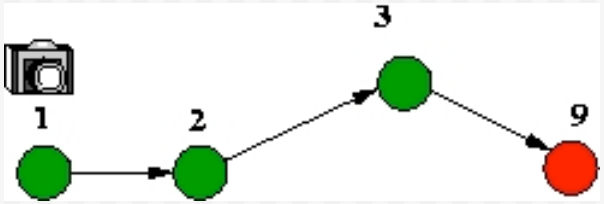
Background: Routing Infrastructure

- Border Gateway Protocol (BGP)
  - Autonomous Systems exchange routing information

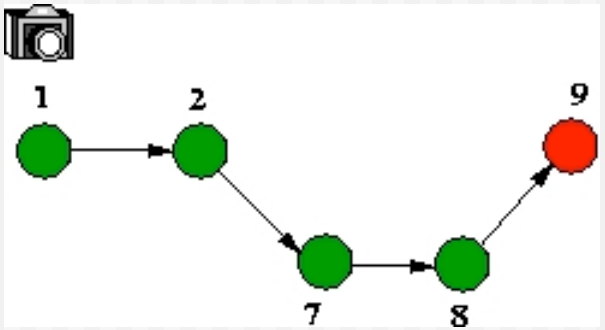


  - B tells A : path to reach D is “D: B,C,D”
  - For update, B tells A: new path to reach D is D: B,E,F,D”

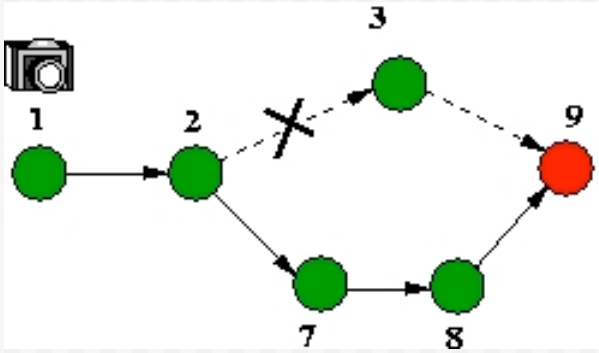
Faults in Networks

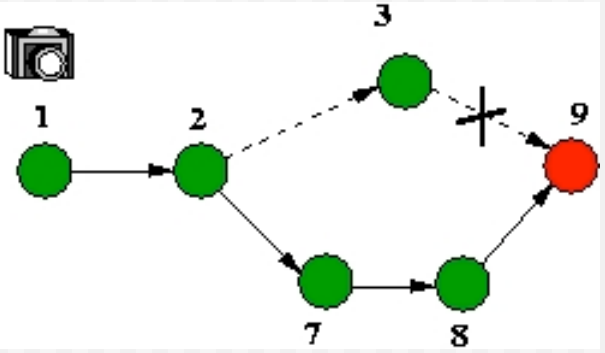


Routing Tree T_0



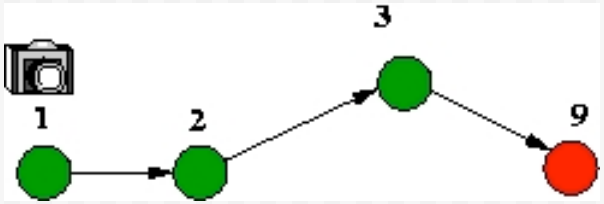
Routing Tree T_1



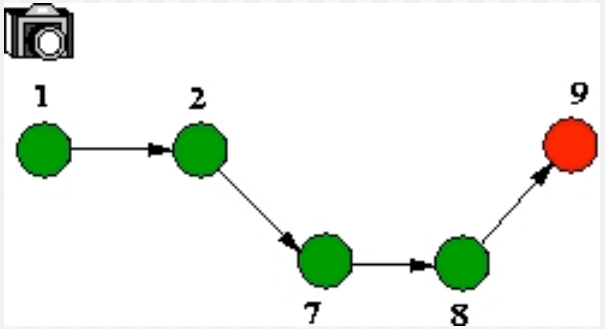


Problem I: Two Different Events could cause same observed effect

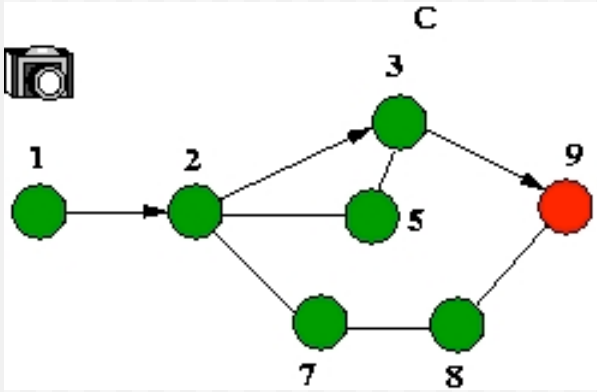
Faults in Networks

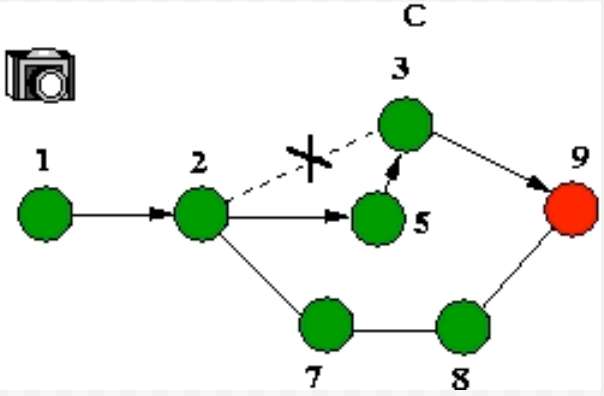


Routing Tree T_0



Routing Tree T_1





Knowing the underlying topology may help eliminate some scenarios, but does not guarantee elimination of problem-I

Challenges in Fault Identification

- Underlying topology not known.
- Path information available from only a select few monitoring points.
- Problem Statement:
  - Given a set of monitoring points M, and incomplete knowledge of the underlying topology, identify possible sets of candidate faults F.
  - PRDC 2004, |F|=1
  - |F| ≥ 1, under submission.

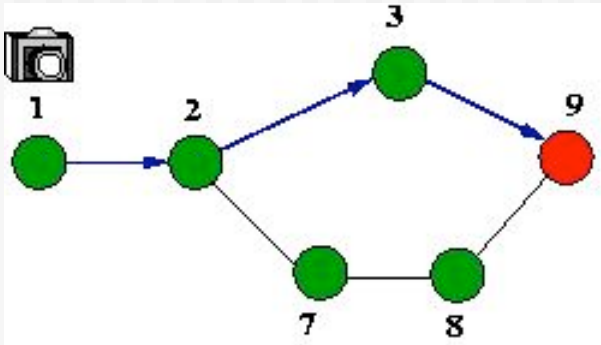
Abstraction

- Internet as graph: (V_N [V_D, E_N [E_D)
  - V_N: AS, E_N: AS peering
  - V_D: prefix/destination
  - E_D: (p,q), $p \in V_D$, $q \in V_N$
- SPVP: Simple Path Vector Protocol
  - Advertise shortest route to all neighbors.

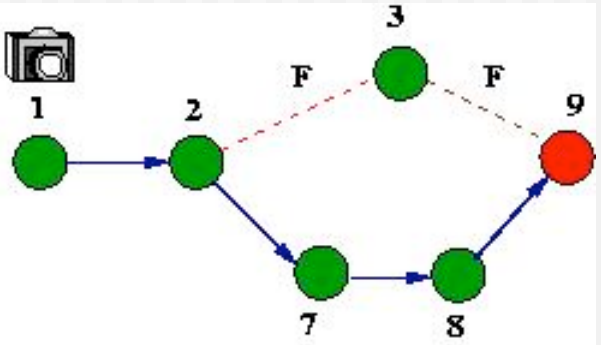
The Basic Approach: From One Monitoring Point

- Let $G=T_0 \cup T_1$
- Let SPT(G,M) be shortest path tree from M to all V_D in G.
- Initial T_0 = SPT (G) = SPT ($T_0 \cup T_1$)
- Final T_1 = SPT (G - F) = SPT ($T_0 \cup T_1$- F), where F is the failed edge, |F|=1.
- Transform T_0 to T_1 by removing the edge closest to M from G.
- FindPath(): returns the path of candidate edges that could have failed.

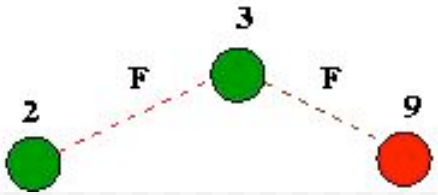
Example



Tree T_0

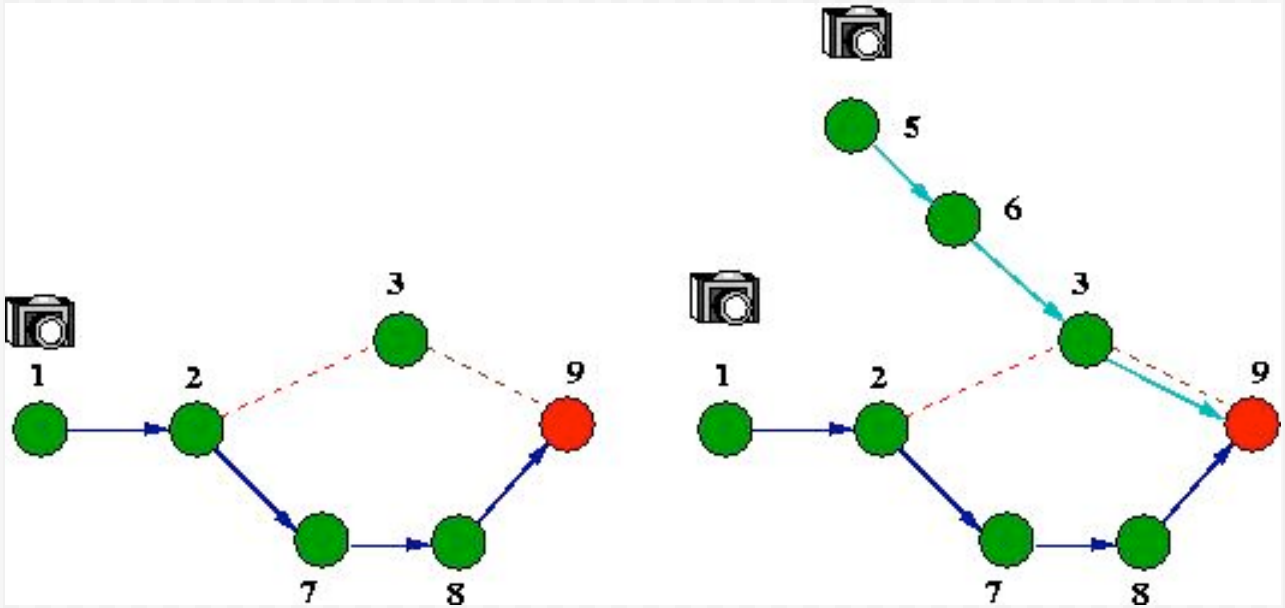


Tree T_1



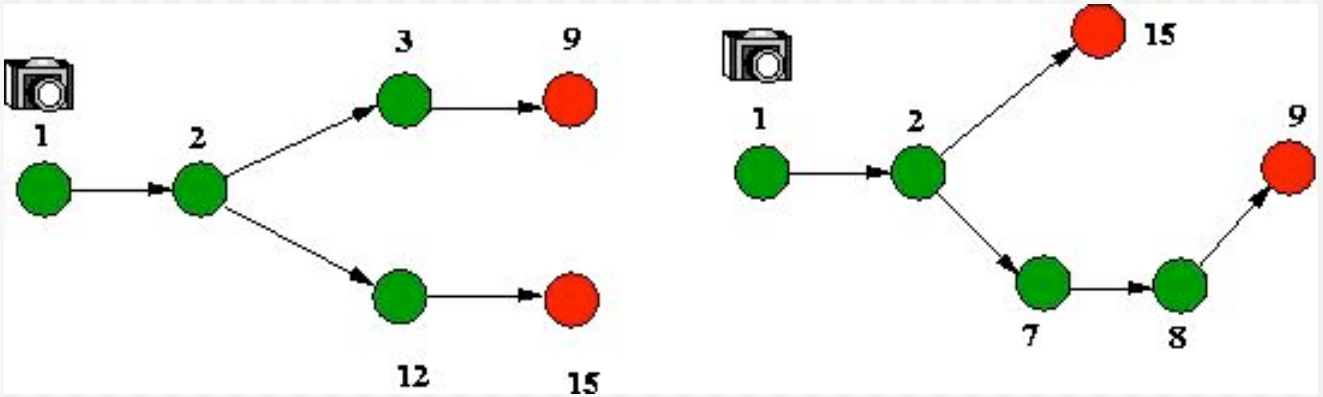
Fault Path from Node 1

Use of Additional Views



- Certain candidate fault edges have to be marked valid. e.g. (3,9)
- Final set of candidate failures = $\cup F_i$ - {valid edges}
- Additional views help, but do not guarantee accurate identification of faults.

Link Additions



T_0 T_1

1. Route to destination 15 gets shorter; an indication of link addition
2. Same algorithm can be applied with the role of T_0 and T_1 reversed
3. Can we deal with a scenarios where one link fails and another is added ?

Example

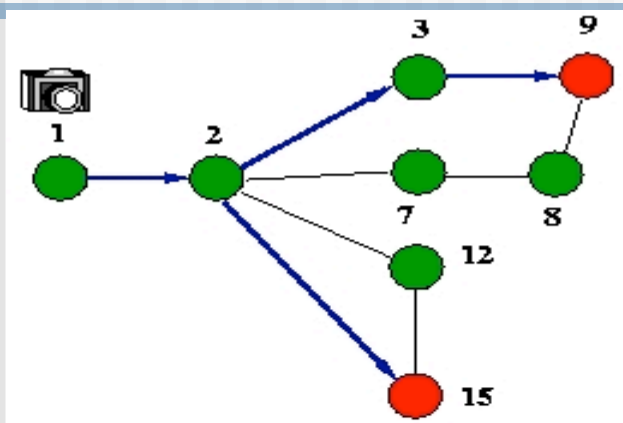


Fig a.

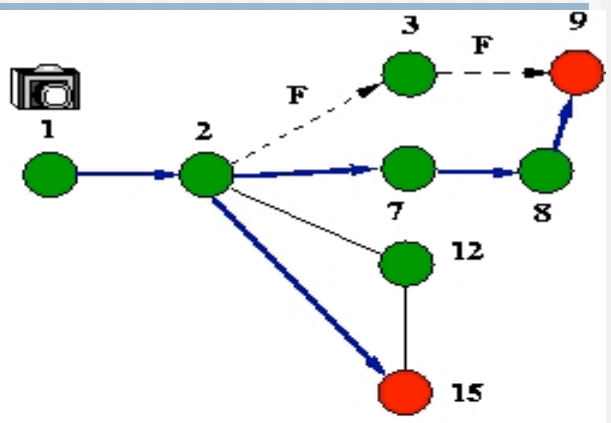


Fig b.

No more links can be removed With original Roles of T_0, T_1

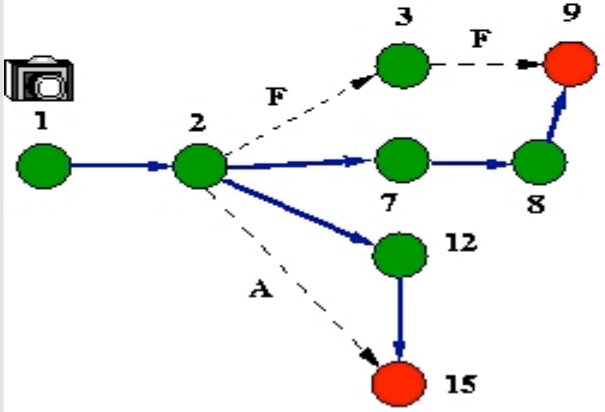
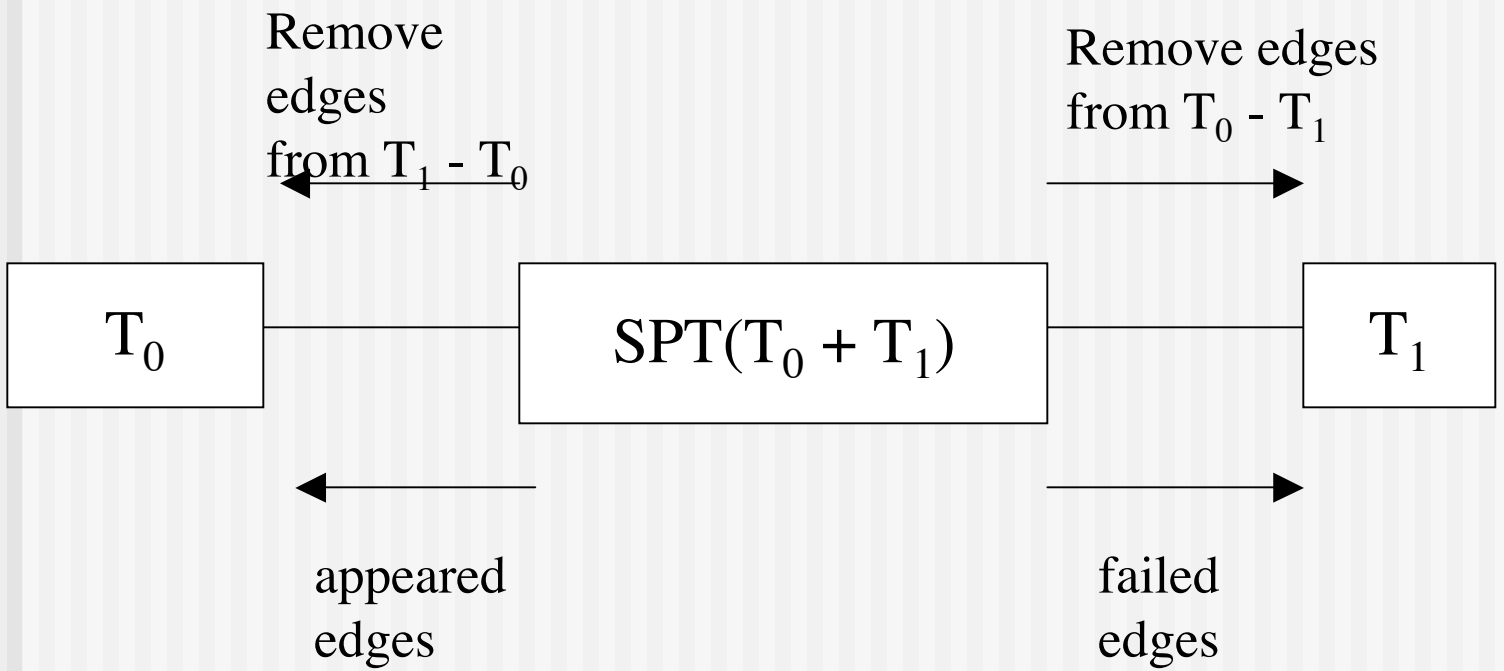


Fig c.

Switch roles of T_0 and T_1
Now, edge (2,15) can be removed.
Thus, edge (2,15) labeled as appeared.
Procedures similar to fault case applied to find out candidate appeared edges.

Summary of Approach



Exploit Symmetry in identification of failed and appeared edges

Final Words

- For a single fault/recovery, our approach can find the candidates for failed or recovered edges.

- Ongoing and Future Work,
  - case of multiple failures (under submission)
  - partial link failures
  - Non-shortest path routing