How to choose a subset of paths from the source node to the destination node in a network for estimating link success probabilities

Soumya Das, Wade Trappe

February 7, 2003

1 Problem Statement

In the network shown below there is one source node and one destination node and that there are M paths from the source node to the destination node. Each node records a random number of packets that pass through it according to some probability associated with the node.We are trying to

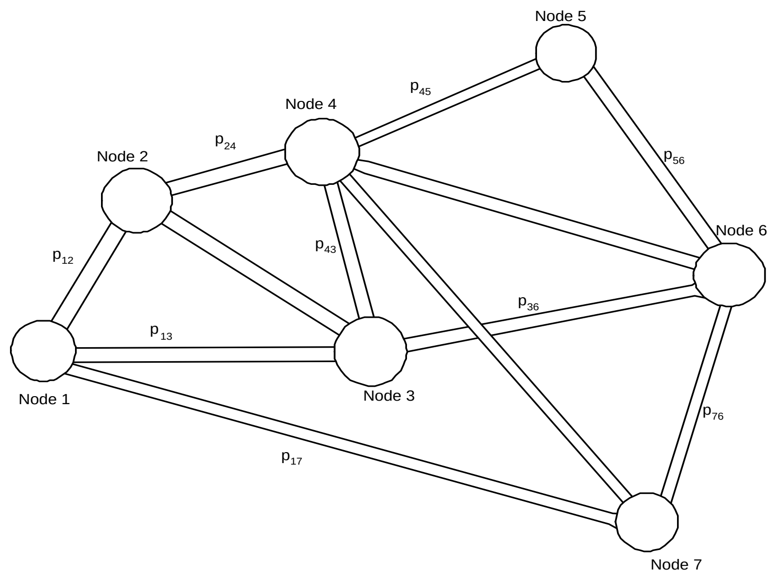

Figure 1: The Network

estimate the success (or failure) probabilities of the links of the network with this information. It is to be observed that when the probability of recording packets is 1 for each of the nodes of the network then this task becomes very simple. However this requires large storage capacities which may not be possible to provide in most of the cases. Now there are two things to be considered here:

- How do we estimate the link success probabilities from the information recorded at the nodes and how close our estimates are to the actual values?

- For finding the above estimates we need to choose some paths out of the M paths.How do we choose the optimum set of paths which gives us a good estimate if not the best estimate? Instead if we consider all the M paths then the computational complexities would increase. Also it is possible to achieve good estimates of link success probabilites by choosing a subset of these M paths.

Here we would be looking only at the second aspect. We have to come up with an algorithm that chooses K of these M paths such that

- We cover as many of the links as possible.
- We have as much overlap as possible.

The above two requirements may be contradictory and so we have to also think how do we resolve the trade-off.

2 Problem Formulation

According to the figure node 1 is the source node and node 6 is the destination node.

We assume here that the routing table is known and all the links are bidirectional. We find there are 6 paths from node 1 to node 6 ($M = 6$).The paths are (1) 1-7-6, (2) 1-3-6, (3) 1-2-3-6,(4) 1-2-4-5-6, (5) 1-2-4-6 and (6) 1-2-4-3-6.

Paths and Links	1-2	1-3	1-7	2-3	2-4	3-4	3-6	4-5	4-6	4-7	5-6	5-7	Weight
(1) 1-7-6	0	0	1	0	0	0	0	0	0	0	0	1	2
(2) 1-3-6	0	1	0	0	0	0	1	0	0	0	0	0	2
(3) 1-2-3-6	1	0	0	1	0	0	1	0	0	0	0	1	3
(4) 1-2-4-5-6	1	0	0	0	1	0	0	1	0	0	0	1	4
(5) 1-2-4-6	1	0	0	0	1	0	0	0	1	0	0	0	3
(6) 1-2-4-3-6	1	0	0	0	1	1	1	0	0	0	0	0	4

Table 1: Table showing links and paths of the network

In the table we have the information of which of the links are lying on a particular path. If a particular link lies on a path then the corresponding cell in the table is represented by a 1 ; otherwise by a 0. The weight column is the number of links a path covers.However we do not need the information in the form of the table and it will be sufficient if we have knowledge about the paths (strings like 1-7-6 , 1-2-3-6 and so on) from the source node to the destination node.We have to choose K_1 of the M paths in such a way that we cover as many of the links of the network. Then of the remaining $(M - K_1)$ paths we have to choose K_2 paths so that we have as much overlap as possible. Thus K depends on our choices of K_1 and K_2 and we have $K = K_1 \cup K_2$

3 Proposed Solution

3.1 Step One

Paths	No. of links covered	Links covered	No. of links not covered by other paths
(1) 1-7-6	2	(1-7,6-7)	2 (1-7,6-7)
(2) 1-3-6	2	(1-3,3-6)	1 (1-3)
(3) 1-2-3-6	3	(1-2,2-3,3-6)	1 (2-3)
(4) 1-2-4-5-6	4	(1-2,2-4,4-5,5-6)	2 (4-5,5-6)
(5) 1-2-4-6	3	(1-2,2-4,4-6)	1 (4-6)
(6) 1-2-4-3-6	4	(1-2,2-4,4-3,3-6)	1 (4-3)

Table 2: Table in step one

We construct the above table from the knowledge of paths from source node to destination node. First we choose the path that has the maximum weight i.e. covers the maximum number of links.In this case, paths (4) and (6) both have the same weight 4. We can use any one of the following methods to choose one of them.They are as follows :

- Flip a coin and decide which path to choose.This is the easier approach.
- Choose that path which covers more number of links that are not covered by other paths.

Path (4) covers links 4-5 and 5-6 that are not covered by any other path, while path (6) covers link 4-3 that is not covered by any other path.This can be found out from the path strings which we have (assuming we don't have the information in tabular format as in Table 1).

So we choose path (4).Thus we have covered 4 links.

3.2 Step Two

Now we consider the remaining paths and find which covers the maximum number of new links (we do not consider the links that are covered by path (4)). The list we get is shown in Table 3:

Paths	No. of new links covered	New links covered	No. of links not covered by other paths
(1) 1-7-6	2	(1-7,6-7)	2 (1-7,6-7)
(2) 1-3-6	2	(1-3,3-6)	1 (1-3)
(3) 1-2-3-6	2	(2-3,3-6)	1 (2-3)
(5) 1-2-4-6	1	(4-6)	1 (4-6)
(6) 1-2-4-3-6	2	(3-4,3-6)	1(3-4)

Table 3: Table in step two

Therefore we see that we can choose any one of (1),(2),(3) or (6) - each of which covers 2 new links.As mentioned earlier, we can decide which path to choose by flipping a coin or rolling a dice (when we have more than two options) or we can choose that path which covers more number of links that are not covered by other paths. The latter method requires additional computation. We choose path (1) because it covers 2 links not covered by other paths (the other three each covers one such link).Thus we have covered $4 + 2 = 6$ links till now.

3.3 Step Three

Now we update the table in a similar way as we did in the previous step.We consider the remaining paths and find which covers maximum number of new links (we consider only those links that are not covered by paths (1) and (4)).We also measure the number of links that are not covered by any other path. The list we get is shown in the following table:

Paths	No. of new links covered	New links covered	No. of links not covered by other paths
(2) 1-3-6	2	(1-3,3-6)	1 (1-3)
(3) 1-2-3-6	2	(2-3,3-6)	1 (2-3)
(5) 1-2-4-6	1	(4-6)	1 (4-6)
(6) 1-2-4-3-6	2	(3-4,3-6)	1 (3-4)

Table 4: Table in step three

Here we find that paths (2),(3) and (6) all are equally well. So we can do either of the two following things.

- We stop here for K_1 such that $K_1 = 2$ and then look for maximizing the number of overlaps.

- We choose another path such that $K_1 = 3$ and then look for maximizing the number of overlaps.

Whichever method we adopt is dependent on what values of K_1 and K_2 we have decided upon. We consider the value of K_1=3 and let us choose path (2) (by rolling a dice).Thus we have covered $4 + 2 + 2 = 8$ links till now.

3.4 Step Four

Now we have to look for the subset K_2 which will maximize the number of overlaps with the paths already chosen. This is done so that our initial estimates of link success probabilities can be refined from additional information.

We choose path (6) because it gives maximum number of overlaps. Thus we have covered $4+2+2+1 = 9$ links and we have 3 overlaps. We stop here with $K_2 = 1$ because the paths (3) and (5) each gives one new link and two overlaps.Thus the paths chosen are (1),(2), (4) and (6). However this does

Paths	No. of overlapping links covered
(3) 1-2-3-6	2 (1-2,3-6)
(5) 1-2-4-6	2 (1-2,2-4)
(6) 1-2-4-3-6	3 (1-2,2-4,3-6)

Table 5: Table in step four

not guarantee the finding of optimum link success probabilities.We can make $K_1 = 2$ and $K_2 = 2$ and may arrive at a different set of paths. It therefore depends on us when we stop looking for more coverage and start looking for overlaps. Therefore the results can be summarised as follows:

- Total number of links in the network = 12.
- Total number of paths from source node to destination node $M = 6$
- Total number of paths chosen for estimation of link success probabilities $= K = K_1 \cup K_2 = 3 \cup 1 = 4$
- Number of links we have covered = 9
- Number of overlapping links amongst chosen paths = 3

Links	Actual link success prob.	Estimated link success prob.	Percentage absolute error
7-6	0.9	0.9	0
1-7	0.93	0.93	0
5-6	0.95	0.88	7.37
4-5-	0.8	0.76	5
2-4	0.9	0.93	3.33
1-2-	0.87	0.9	3.45
3-6	0.96	0.97	1.04
1-3	0.85	0.84	1.17
4-3	0.98	0.98	0

Table 6: Table showing estimated link success probabilites

4 Simulation Results

The estimated link success probabilities (obtained from simulations in MATLAB) are given in tabular form above. The paths (1),(2),(4) and (6) were used to calculate the link success probabilites.The simulations were carried out with N =Number of packets= 40 and R =Number of runs= 5.

The mean absolute percentage error is 2.37.This error decreases when we consider more number of packets or runs in our simulations.