Algorithm Design for Multi-resolution Overlay Measurement

Yong Liu

1 Introduction

2 Overlay Probing Pair Deployment

We model a underlay network as a directed graph $G = (V, E)$, where V is the set of routers, and E is the set of underlay links. On top of the underlay network G, we build an overlay network by attaching overlay boxes to a subset of routers to implement new network functions/services without changing the existing architecture/operation of the underlay network. Let $V^o \subset V$ be the set of overlay boxes installed. For each overlay box $i \in V^o$, let $u(i) \in V$ be the underlay router to which i is attached. A logical overlay link l from overlay box i to overlay box j corresponds to the underlay routing path p_l from underlay router $u(i)$ to $u(j)$. Let E^o be the set of *overlay links* between overlay boxes, then the overlay network is represented by an overlay graph $G^o = (V^o, E^o)$.

To facilitate overlay routing, overlay boxes send probing traffic to each other to measure the status of the underlay network. We need to determine which overlay pairs should participate in the measurement to maximally recover the status of underlay links with controlled overlay measurement overhead.

2.1 Independent Probing Pairs

Let x_e be the delay on underlay link $e \in E$, let y_l be the measured delay on overlay link $l \in E^o$, let A be the routing matrix for all the overlay links such that $A[l, e] = 1$ if underlay link e is on the underlay path p_l for overlay link l, $A[l, e] = 0$ otherwise. Since the end-to-end delay on overlay link l is the summation of delays on all links along the corresponding underlay path p_l, then we have

$$AX = Y,$$

where $X = [x_1, \cdots, X_{|E|}]^T$ is the underlay link delay vector to be inferred, and $Y = [y_1, \cdots, y_{|E^o|}]^T$ is the measured overlay link delay vector. In order to fully recover X, we want to choose a subset of overlay pairs such that their corresponding rows in the routing matrix A form a sub-matrix with full rank of $|E|$.

Given all possible overlay box locations V^o, we can construct a fully connected overlay graph, and the corresponding routing matrix for all the overlay links is A^F. Suppose the row rank of A^F is k, we can use the classical orthogonal-triangular matrix decomposition to find k independent rows of A^F. Each row corresponds to an overlay link, i.e., a probing pair between overlay nodes. We can simply choose those k overlay pairs to conduct delay probing. And the delay on any other overlay link (probing pair) is linearly dependent on the delays of those k overlay links. Therefore, theoretically there is no need to conduct delay measurement beyond those k overlay pairs.

Note that if $k < |E|$, it is impossible to recover delays on all underlay links. One can either install more overlay boxes on more routers, or simply estimate delays on virtual links, each of which consists of the sum of delays of several underlay links.

2.2 Underlay Link Coverage

For our low frequency probing, the main goal is not to recover delays on all underlay links. Instead, the goal is to quickly identify potential bottlenecks. Specifically, let E^m be the set of underlay links to be monitored by overlay measurement. For each underlay link $e \in E^m$, let n_e be the target number of overlay probings to be applied to e, and L_e be the set of overlay links traversing e, i.e.,

$$L_e = \{l \in E^o : A[l, e] = 1\}.$$

We now need to choose a subset of overlay links to conduct probing such that all underlay links in E^m can be probed by their target numbers, and the total overlay measurement overhead is minimized.

We can formulate this problem as a binary programming problem. Let w_l be the measurement overhead along overlay link l, and x_l be the binary variable to indicate whether or not to include overlay link l in the measurement. Then we want to cover all the underlay links with the minimal overlay measurement overhead.

$$\min_{\{x_l : l \in E^o\}} \textstyle\sum_{l \in E^o} x_l w_l, \tag{1}$$

$$\text{subject to:} \quad \textstyle\sum_{l \in L_e} x_l \geq n_e, \quad \forall e \in E^M. \tag{2}$$

When the network is small, we can get the exact optimal overlay probing pairs by using integer programming tools. If the network is too large, we can develop heuristic algorithms to quickly obtain close-to-optimal solutions. Specifically, when $w_l = c$, $\forall l$ and $n_e = 1$, $\forall e \in E^M$, in other words, all links only need to be covered once, and all measurement pairs have the same overhead, the problem is exactly the classical set cover problem, we can adopt a greedy heuristic algorithm to solve it, see Algorithm 1.

Algorithm 1 Greedy Algorithm for Uniform Underlay Link Coverage with Homogeneous Overlay Measurement Overhead

INPUT: underlay links to be covered E^m; candidate overlay probing pairs E^o;
OUTPUT: selected overlay probing pairs S;
$S = \emptyset$;
while $E^m \neq \emptyset$ and $E^o \neq \emptyset$ **do**
  $score(l) = |p_l \cap E^m|, \forall l \in E^o$;
  find an overlay link l with the highest score;
  if $score(l) == 0$ **then**
    break;
  end if
  $S = S \cup \{l\}$, $E^o = E^o - \{l\}$; $E^m = E^m - p_l$;
end while
return S;

If $n_e = 1, \forall e \in E^m$, but w_l are not homogeneous, in other words, all links only need to be covered once, but overlay pairs have heterogeneous measurement overhead, it is essentially the weighted set cover problem, we can use see Algorithm 2 to quickly obtain an approximate solution.

In the most general case where different links need to be covered by different numbers of times, and overlay pairs have heterogeneous measurement overheads, we will develop another heuristic algorithm as illustrated in Algorithm 3. We basically run the (weighted) greedy heuristic algorithm, but at each step, whenever an overlay link with the highest score is chosen, we reduce the count of each uncovered underlay link on its path by one, remove an underlay link if its counter goes down to zero.

2.3 Numerical Results

For the network illustrated in Figure 1, there are eight routers, among them, the outside routers are attached with overlay boxes. If all links are to be covered exactly once, we can run the Algorithm 1 to select overlay links for measurement. In the first round, the algorithm will choose one overlay link, e.g., AD, to cover three underlay links, then only four underlay links left to be covered. Then the

Algorithm 2 Greedy Algorithm for Uniform Underlay Link Coverage with Heterogeneous Overlay Measurement Overhead

INPUT: underlay links to be covered E^m; candidate overlay probing pairs E^o; measurement overhead w_l on each overlay link $l \in E^o$;
OUTPUT: selected overlay probing pairs S;
$S = \emptyset$;
while $E^m \neq \emptyset$ and $E^o \neq \emptyset$ **do**
  $score(l) = \frac{|p_l \cap E^m|}{w_l}$;
  find an overlay link l with the highest score;
  if $score(l) == 0$ **then**
    break;
  end if
  $S = S \cup \{l\}$, $E^o = E^o - \{l\}$; $E^m = E^m - p_l$;
end while
return S;

Algorithm 3 Greedy Algorithm for Weighted Underlay Link Coverage with Heterogeneous Overlay Measurement Overhead

INPUT: underlay links to be covered E^m; candidate overlay probing pairs E^o; measurement overhead w_l on each overlay link $l \in E^o$; target probing number n_e on each underlay link $e \in E^m$;
OUTPUT: selected overlay probing pairs S; the number of probing times m_s on each selected pair $s \in S$;
$S = \emptyset$; $m_s = 0, \forall s \in S$;
while $E^m \neq \emptyset$ and $E^o \neq \emptyset$ **do**
  $score(l) = \frac{|p_l \cap E^m|}{w_l}$;
  find an overlay link l with the highest score;
  if $score(l) == 0$ **then**
    break;
  end if
  $S = S \cup \{l\}, m_l + +$;
  $C = E^m \cap p_l$;
  for $\forall e \in C$ **do**
    if $- - n_e = 0$ **then**
      $E^m = E^m - \{e\}$;
    end if
  end for
end while
return S and $\{m_s, s \in S\}$;

algorithm will choose BF and CE to cover all the underlay links, then the solution is $\{AD, BF, CE\}$.

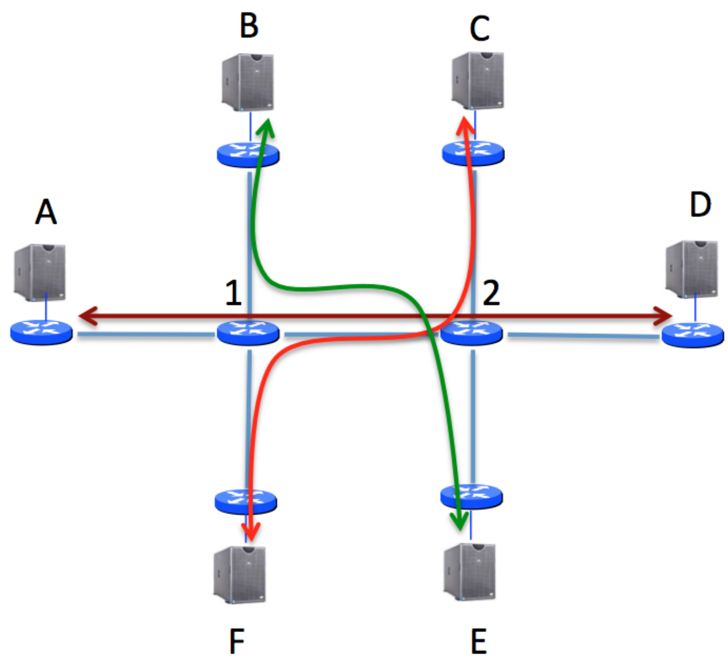

Figure 1: Example Overlay and Underlay Topology

Now if we change the requirement, and the link between router 1 and 2 needs to be covered at least twice. The algorithm will first choose a three-hop overlay link, e.g., AD, but now link 12 still need to be covered another time. Then the algorithm will choose either BE or CF to cover three more underlay links. Finally, the algorithm will choose another overlay link to cover the remaining two links. So the final solution is $\{AD, BE, CF\}$, and indeed the link between router 1 and 2 are covered three times.

3 Underlay Bottleneck Inference

After delay probing along the chosen overlay links, we can clarify the status of each overlay link as normal/good or abnormal/bad. Let v_l be the state variable of overlay link l, with $v_l = 1$ if the overlay link is normal, and $v_l = 0$ if the overlay link is abnormal. We want to infer the status of each monitored underlay links based on the measured overlay link statuses. Let u_e be the state variable of underlay link e, with $u_e = 1$ if the underlay link is normal, and $u_e = 0$ if the underlay link is abnormal. Here is the assumption for the inference:

- rule 1: if a path is good, all links along the path is good
- rule 2: if a link is bad, all paths traversing the link are bad

Those rules can be translated into the following equation:

$$v_l = \prod_{e \in p_l} u_e, \forall l \in E^o. \quad (3)$$

After overlay probing, let L^p be the set of abnormal overlay links, $L^p = \{l \in E^o : v_l = 0\}$, and L^n be the set of normal overlay links, $L^n = \{l \in E^o : v_l = 1\}$. We want to infer the statuses of the underlay links $\{u_e, e \in E\}$ based on L^p and L^n.

3.1 Inference Model

Based on (3), it is safe to mark all underlay links on normal overlay links as normal. Let $G = \underset{l \in L^n}{\cup} p_l$ be the set of underlay links traversed by at least one normal overlay link. Then it is safe to mark underlay links in G as normal, i.e., we set $u_e = 1$, $\forall e \in G$. Underlay links that are not on any normal overlay link are the probable performance bottlenecks. One naive way is to mark all those underlay links as potential bottlenecks, i.e., $u_e = 0, \forall e \in E - G$.

But this classification is too crude, might generate too many probable bottlenecks for high resolution probing. Given that in practice congestion only happens with low probability, we can choose underlay link status assignment that maximizes the *likelihood* of the measured overlay link statuses, i.e, **the Maximum Likelihood Estimation**, or we want to estimate the congestion probability of individual underlay links given the overlay link statuses using **the Bayesian Inference Model**.

Specifically, given the measured binary vector of overlay link statuses $V = \{v_l, l \in E^o\}$, there are many possible underlay link status vectors $U = \{u_e, e \in E\}$ that can "explain" the observed overlay link statuses, i.e., satisfy the constraint (3). Let $\mathcal{U}(V)$ be the set of underlay link status vectors satisfying (3) for a given

V. Suppose α_e is the probability that underlay link e experiences performance problems, (this can be obtained from history data), then the likelihood of underlay link status vector U can be calculated as:

$$P\left(U = \{u_e, e \in E\}\right) = \prod_{e \in E} \alpha_e^{1-u_e}(1-\alpha_e)^{u_e}. \tag{4}$$

We can define the logarithm of the likelihood as:

$$\begin{aligned} \mathcal{L}\left(U = \{u_e, e \in E\}\right) &= \log \prod_{e \in E} \alpha_e^{1-u_e}(1-\alpha_e)^{u_e} && (5) \\ &= \sum_{e \in E} \left((1-u_e)\log \alpha_e + u_e \log(1-\alpha_e)\right) && (6) \\ &= \sum_{e \in E} \left(u_e \log(\frac{1}{\alpha_e} - 1) + \log \alpha_e\right) && (7) \end{aligned}$$

3.2 Maximum Likelihood Estimation

For a given observed overlay link status vector V, we want to choose the most likely underlay link status vector $U \in \mathcal{U}(V)$. Therefore our inference problem can be expressed as:

$$U^*(V) = \underset{U \in \mathcal{U}(V)}{\mathbf{argmax}} \quad \mathcal{L}(U). \tag{8}$$

By plugging in (3) and (7) into (8), the inference problem can be formulated as a binary programming problem:

$$\begin{aligned} &\max_{\{u_e : e \in E\}} \textstyle\sum_{e \in E} \left(u_e \log(\frac{1}{\alpha_e} - 1) + \log \alpha_e\right), \\ \text{subject to:} \quad &\textstyle\prod_{e \in p_l} u_e = v_l, \quad \forall l \in E^o \end{aligned}$$

It is easy to show that the problem is equivalent to the following problem:

$$\begin{aligned} &\max_{\{u_e : e \in E\}} \textstyle\sum_{e \in E} \left(u_e \log(\frac{1}{\alpha_e} - 1) + \log \alpha_e\right), \\ \text{subject to:} \quad &\textstyle\sum_{e \in p_l} (1-u_e) \geq 1, \quad \forall l \in L^p \\ &u_e = 1, \quad \forall e \in G = \bigcup_{l \in L^n} p_l \end{aligned}$$

By focusing on only unmarked links in $E - G$, and set $m_e = 1 - u_e$, the inference problem can be reduced to an equivalent binary optimization problem

$$\min_{\{m_e : e \in E-G\}} \textstyle\sum_{e \in E-G} m_e \log(\frac{1}{\alpha_e} - 1), \tag{9}$$

$$\text{subject to:} \quad \textstyle\sum_{e \in p_l \cap (E-G)} m_e \geq 1, \quad \forall l \in L^p, \tag{10}$$

Similar to the coverage problem in (1), it can be solved by some binary programing tool for a small network. When the network size is large, we can develop heuristic algorithms to obtain close-to-optimal solutions.

3.2.1 Heuristic Algorithms

1. No History Data or Homogeneous Prior Link Congestion Probability

If there is no history data about link congestion probability, or link congestion probability are homogeneous, we can assume $\alpha_e = c$, $\forall e \in E - G$. If $\alpha_e > 0.5$, i.e., links are more likely to be congested than not congested, then the coefficients $\log(\frac{1}{\alpha_e} - 1)$ in (9) are negative. The optimal solution is simply $m_e = 1, \forall e \in E - G$, i.e., classifying all unmarked links as congested. If $\alpha_e < 0.5$, then $\log(\frac{1}{\alpha_e} - 1) > 0$, then the optimization problem in (9) is equivalent to

$$\min_{\{m_e : e \in E-G\}} \sum_{e \in E-G} m_e,$$
$$\text{subject to:} \quad \sum_{e \in p_l \cap (E-G)} m_e \geq 1, \quad \forall l \in L^p,$$

which is essentially a set cover problem, with the set be the set of abnormal overlay links L^p, and each unmarked underlay link $e \in E - G$ corresponds to the subset $L^p \cap L_e$. We can develop a greedy algorithm as illustrated in Algorithm 4 to solve it.

For the example in Figure 1, if we deploy three overlay link measurement {AD, BE, CF}, here is some example inference scenarios:

1. If all overlay measurement are normal, then all underlay links are marked as normal;
2. If AD and BE are normal, but CF is abnormal, then the algorithm will mark either F1 or C2 as abnormal, and mark all other links as normal;
3. If AD is normal, but CF and BE are abnormal, then the algorithm will first mark A1, 12 and D2 as normal, but will mark either B1 or E2 as abnormal, and mark either F1 or C2 as abnormal.
4. If all overlay links are abnormal, then the algorithm will mark link 12 as abnormal, and mark all other links as normal.

2. Heterogeneous Prior Link Congestion Probability

With heterogeneous prior link congestion probability α_e, we need to solve (9) as a weighted set cover problem. With the exception that for links with $\alpha_e > 0.5$, their weights $\log(\frac{1}{\alpha_e} - 1) < 0$, to minimize the weighted sum objective function in (9), those links should be marked as abnormal, i.e., set $u_e = 0$. Then we can

Algorithm 4 Greedy Algorithm for Underlay Link Status Inference without History Data or Homogeneous Link Congestion Probability of $\alpha_e < 0.5$

INPUT: binary overlay link status vector $V = \{v_l, l \in E^o\}$;
OUTPUT: binary underlay link status vector $U = \{u_e, e \in E\}$;
$G = \emptyset$; $L^p = \emptyset$;
Initialize $u_e = 1, \forall e \in E$;
for $\forall l \in E^o$ **do**
  if $v_l == 1$ **then**
    $G = G \cup p_l$;
  else
    $L^p = L^p \cup \{l\}$
  end if
end for
while $L^p \neq \emptyset$ and $G \neq E$ **do**
  $score(e) = |L_e \cap L^p|, \forall e \in E - G$;
  find underlay link $e \in E - G$ with the highest score;
  if $score(e) == 0$ **then**
    return ERROR;
  end if
  $u_e = 0, G = G \cup \{e\}, L^p = L^p - L_e$;
end while
return U;

infer the statuses of the remaining unmarked links using the heuristic algorithm presented in Algorithm 5

Algorithm 5 Greedy Algorithm for Underlay Link Status Inference with Heterogeneous Link Congestion Probability

---

INPUT: binary overlay link status vector $V = \{v_l, l \in E^o\}$;
OUTPUT: binary underlay link status vector $U = \{u_e, e \in E\}$;
$G = \emptyset$; $L^p = \emptyset$;
Inititalize $u_e = 1, \forall e \in E$;
for $\forall l \in E^o$ **do**
  if $v_l == 1$ **then**
    $G = G \cup p_l$;
  else
    $L^p = L^p \cup \{l\}$
  end if
end for
for $\forall e \in E - G$ **do**
  if $\alpha_e > 0.5$ **then**
    $u_e = 0, G = G \cup \{e\}, L^p = L^p - L_e$
  end if
end for
while $L^p \neq \emptyset$ and $G \neq E$ **do**
  $score(e) = \frac{|L_e \cap L^p|}{\log(\frac{1}{\alpha_e} - 1)}, \forall e \in E - G$
  find underlay link $e \in E - G$ with the highest score;
  if $score(e) == 0$ **then**
    return ERROR;
  end if
  $u_e = 0, G = G \cup \{e\}, L^p = L^p - L_e$;
end while
return U;

For the example in Figure 1, again if we deploy three overlay link measurement $\{$AD, BE, CF$\}$, but we set prior congestion probability as the following: $\alpha_{C2} = \alpha_{B1} = \alpha_{D2} = 0.1$, and $\alpha_{A1} = \alpha_{E2} = \alpha_{F1} = 0.05$, for the same set of scenarios as in the homogeneous setting, we will obtain different results:

1. If all overlay measurement are normal, then all underlay links are marked as normal;
2. If AD and BE are normal, but CF is abnormal, then the algorithm will mark C2 as abnormal, and mark all other links as normal;

3. If AD is normal, but CF and BE are abnormal, then the algorithm will mark C2 and B1 as abnormal, and mark all other links as normal;

4. If all overlay links are abnormal, if $\alpha_{12} = 0.01$, then the algorithm will mark link 12 as abnormal, and all other links are normal; but if $\alpha_{12} = 0.00001$, then the algorithm will mark link C2, B1 and D2 as abnormal, and all other links are normal.

3.3 Bayesian Inference Model

Given history data, we can calculate the prior distribution of the congestion probability on each underlay link e as $P(u_e = 0) = \alpha_e, \forall e \in E$. After the overlay measurement result $V = \{v_l, l \in E^o\}$, we want to update the the congestion probability on overlay links as $P(u_e = 0|V)$. Similar to the MLE, according to the rule in (3), for each underlay link $e \in G = \underset{l \in L^n}{\cup} p_l$, we know for sure it is normal, thus we set $u_e = 1$. But for underlay links in $E - G$, we don't know for sure their real statuses. Instead we can calculate their congestion probability using the following Bayesian formula:

$$
\begin{aligned}
P(u_e = 0|V) &= \frac{P(u_e = 0, V)}{P(V)} && (11)\\
&= \frac{P(U \in \mathcal{U}(V), u_e = 0)}{P(U \in \mathcal{U}(V))} && (12)\\
&= \frac{\sum_{U \in \mathcal{U}_e(V)} P(U)}{\sum_{U \in \mathcal{U}(V)} P(U)}, && (13)
\end{aligned}
$$

where $\mathcal{U}(V) = \{U : u_i = 1, \forall i \in G, \text{ and } \prod_{i \in p_l \cap (E-G)} u_i = 0, \forall l \in L^p\}$ is the set of all possible underlay link status vectors that can explain the observed overlay link statuses V, and $\mathcal{U}_e(V) = \{U : u_i = 1, \forall i \in G; u_e = 0; \text{ and } \prod_{i \in p_l \cap (E-G-\{e\})} u_i = 0, \forall l \in L^p - L_e\}$ is the set of all possible underlay link status vectors that can explain the observed overlay link statuses and underlay link e is marked as congested, and $P(U)$ can be calculated using (4).

After calculating the posterior link congestion probability, we can set a probability threshold η, and mark all underlay links with $P(u_e = 0|V) \geq \eta$ as probable bottlenecks for next round probing.

For the example in Figure 1, if all three overlay links are abnormal, and the prior congestion probabilities are α_{12}, $\alpha_{C2} = \alpha_{B1} = \alpha_{D2} = 0.1$, and $\alpha_{A1} =$

$\alpha_{E2} = \alpha_{F1} = 0.05$, then the posterior congestion probability on link 12 is

$$\begin{aligned} & P(u_{12} = 0 | \text{AD, BE, CF are abnormal}) \\ = & \frac{\alpha_{12}}{\alpha_{12} + (1-\alpha_{12})(1-(1-0.1)(1-0.05))^3} \\ = & \frac{\alpha_{12}}{0.0030486 + 0.9969514\alpha_{12}} \end{aligned}$$

If $\alpha_{12} = 0.01$, the posterior congestion probability on link 12 is 0.6947, and if $\alpha_{12} = 0.00001$, the posterior congestion probability on link 12 is 0.00327.

3.4 Overlay Measurement Errors

So far, we have been assuming that the overlay measurement can accurately identify the normal/abnormal status on each overlay link. In practice, overlay measurement might also experience errors: a normal overlay link might be identified as an abnormal overlay link, i.e., **false positive**; or an abnormal overlay link might be identified as a normal overlay link, i.e., **false negative**. Let $\hat{v}_l$ be the measured status, and v_l be the actual status on overlay link l.

Then the false positive rate on overlay link l is:

$$F_l^p = P(\hat{v}_l = 0 | v_l = 1).$$

And the false negative rate on overlay link l is:

$$F_l^n = P(\hat{v}_l = 1 | v_l = 0).$$

Underlay link status inference should take into account overlay measurement error probability. More generally, let $\hat{V}$ be the measured overlay link status vector and V be the actual overlay link status vector. The measurement error model can be characterized by the conditional probability of $P(\hat{V}|V)$ and $P(V|\hat{V})$. Then we can update our underlay link status inference results in the following way.

3.4.1 Updated MLE Inferences

Given the measured overlay link status vector $\hat{V}$, we can calculate the set of all possible real overlay link status vectors as $\mathcal{V}(\hat{V})$, and their associated probability $P(V|\hat{V})$, $\forall V \in \mathcal{V}(\hat{V})$. Using algorithms presented in Section 3.2, we can obtain MLE underlay link status estimation for each V, and get the underlay link status vector $U^*(V)$ as defined in (8). Then the final underlay link congestion probabilities can be estimated as

$$M(\hat{V}) = I - \hat{U}(\hat{V}) = \sum_{V \in \mathcal{V}(\hat{V})} (I - U^*(V)) P(V|\hat{V}), \qquad (14)$$

where I is the column vector with all elements being 1.

3.4.2 Updated Bayesian Inference Model

With the measurement error model, we can also update the Bayesian inference model presented in Section 3.3. Specifically, given the measured overlay link status vector $\hat{V}$, we can calculate the posterior congestion probability for underlay link e as:

$$
\begin{aligned}
P(u_e = 0|\hat{V}) &= \frac{P(u_e = 0, \hat{V})}{P(\hat{V})} && (15) \\
&= \frac{\sum_{\{U: u_e=0\}} P(U)P(\hat{V}|g(U))}{\sum_{\{U\}} P(U)P(\hat{V}|g(U))} && (16) \\
&= \frac{\sum_{U \in \mathcal{F}_e(\hat{V})} P(U)P(\hat{V}|g(U))}{\sum_{U \in \mathcal{F}(\hat{V})} P(U)P(\hat{V}|g(U))}, && (17)
\end{aligned}
$$

where $g(U)$ is the overlay link status vector if the underlay link status vector is U; $\mathcal{F}(\hat{V}) = \{U : g(U) \in \mathcal{V}(\hat{V})\}$ is the set of underlay link status vectors whose corresponding overlay link status vectors can be measured as $\hat{V}$; and $\mathcal{F}_e(\hat{V}) = \{U : u_e = 0 \text{ and } g(U) \in \mathcal{V}(\hat{V})\}$ is the set of underlay link status vectors whose corresponding overlay link status vectors can be measured as $\hat{V}$ and underlay link e is marked as congested.

3.4.3 Error Probability Calculation

If we assume measurement errors on all overlay links are independent, then we can calculate the error probability as:

$$
\begin{aligned}
P(\hat{V}|V) &= \prod_{l \in E^o} P(\hat{v}_l|v_l); \\
P(V|\hat{V}) &= \prod_{l \in E^o} P(v_l|\hat{v}_l).
\end{aligned}
$$

$P(\hat{v}_l|v_l)$ can be directly calculated from the false positive and false negative rate F_l^p and F_l^n. $P(v_l|\hat{v}_l)$ can be calculated as

$$
P(v_l|\hat{v}_l) = \frac{P(v_l, \hat{v}_l)}{P(\hat{v}_l)} = \frac{P(\hat{v}_l|v_l)P(v_l)}{\sum_{v_l=0,1} P(\hat{v}_l|v_l)P(v_l)}, \tag{18}
$$

where $P(v_l)$ can be obtained from the prior underlay link congestion probability $P(u_e = 0), \forall e \in p_l$.

In a practical setting, we can assume only a small number of overlay measurement errors can happen at the same time:

$$P(V|\hat{V}) = P(V|\hat{V}) = 0, \quad \text{if } \|V - \hat{V}\| > \kappa. \tag{19}$$

If we set $\kappa = 1$, it means we only consider at most one overlay link status is wrongly classified. Then $\mathcal{V}(\hat{V})$ only includes overlay link status vectors that have at most one element different from the measured vector $\hat{V}$.