Gleaning Network-Wide Congestion Information from Packet Markings

Florin Dinu T. S. Eugene Ng
Rice University

ABSTRACT

Distributed control protocols routinely have to operate oblivious of dynamic network information for scalability or complexity reasons. However, more informed protocols are likely to make more intelligent decisions. We argue that protocols can leverage dynamic congestion information without suffering the mentioned penalties. In this paper we show that routers can readily exchange congestion information in a purely passive fashion using congestion markings from existing traffic. As a result, each router can locally infer a congestion map of the network. Moreover, the maps are continuously updated with near real-time information. Our solution for building the congestion maps leverages standardized and widely used congestion management protocols and does not require changes to end hosts. We find that 90% of the time, the inference accuracy is usually within 10% even for environments with multiple congestion points and sudden changes in the traffic pattern.

1. INTRODUCTION

Distributed control protocols have to routinely make decisions without the help of dynamic network information. In many cases this is due to scalability concerns, the burden of introducing additional complexity or the need to maintain backward compatibility. However, having more informed network protocols can lead to more intelligent decisions. Our novel proposition is that dynamic congestion information can be gathered without the mentioned drawbacks. Leveraging dynamic congestion information needs to consider challenges such as the potential for oscillations or instability which are inherent in using dynamic data. However, this behavior is application dependent. In this paper we focus on efficiently and accurately distributing dynamic router level congestion information in the network and using it for inferring network-wide congestion information.

The existence of congestion is an important indicator of the health of a network. Congestion indicates a severe degradation of the service level provided by the network and even a possible danger to the stability of the basic routing functionality [21]. Several distributed router level algorithms can benefit from dynamic congestion information. For example, IP fast-reroute algorithms [16, 15, 22] can leverage congestion information when computing and choosing backup paths or if they monitor the effect of their routing decision. Automated verification mechanisms that use multiple vantage points to verify the correct functioning of routers [19, 13, 24] can use congestion information to reason whether packet loss was caused by congestion or by errors in protocol implementations. Distributed traffic engineering algorithms [12] can leverage congestion information to balance the load in the network.

Any solution designed to provide dynamic congestion information to distributed router level algorithms must satisfy a number of important requirements. First, solutions must scale with respect to both the network size and the congestion information update rate. Second, solutions must be robust to various network conditions. For example, the presence of multiple congested routers should have little impact. Third, solutions should be accurate. Today, accurate results can be obtained if congestion information is periodically retrieved from routers, using a network management protocol. Unfortunately, these methods are neither scalable nor robust. They are not scalable for large network sizes or high update rates because these scenarios significantly increase the overhead the solutions add to the network. These methods are not robust because during congestion periods messages can get lost or suffer from multiple timeouts. The lack of a scalable and robust method for distributing dynamic congestion information to routers is an important shortcoming that could reduce the effectiveness and the adoption likelihood of useful network level algorithms. The absence of such a method often forces researchers to make unrealistic assumptions that completely ignore the effects of congestion. For example, IP fast re-route algorithms are not currently concerned with congestion [16, 22] even though it is well known that their decisions affect and are affected by the congestion in the network [22].

In this paper we show that a network-wide congestion map can be inferred with no network traffic overhead, minimal infrastructure changes and little demand on routers by leveraging congestion marking from existing traffic. The key insight in obtaining a network-wide congestion map is combining several standardized and widely used protocols. First, explicit congestion notification (ECN) and active queue man-

agement (AQM) algorithms allow the inference of aggregate, path level congestion severity information. We use marking probability as a measure of congestion severity. Subsequently, routing information can be used to break down the path level information to obtain detailed, link level congestion location and marking probability. In practice, several factors can cause measurement inaccuracies. We construct an analytical model that quantifies the effect of the most important factor, the method receivers used to echo congestion markings back to sources. Based on the analytical model, we derive two solutions for obtaining congestion estimates.

Our solutions are scalable. Extra traffic is never added into the network and the demand on the routers is low. Moreover, we show that good results can be obtained even for update intervals of only a few seconds. The reason is the reliance on existing traffic which allows the maps to be continuously updated with near real-time information. The solutions are robust. We test our solutions with simulations on network environments with several congestion points and sudden variations in the traffic pattern. We are able to show that even in these scenarios our solutions can infer congestion multiple hops away with good accuracy: 90% of the time the inference accuracy is usually within 10%. We also show that our solutions are resilient to other factors such as packet loss and variable transmission delays. Also, our solutions can be incrementally deployed and do not require changes to either current protocols or to the end-hosts.

The rest of the paper is organized as follows. §2 describes our assumptions and an overview of the solution. §3 presents the process of inferring aggregate path level congestion estimates. §4 discusses the construction of the congestion maps and their properties. In §5 we evaluate the accuracy of our approach. We discuss additional issues in §6 and present related work in §7. We finally conclude in §8.

2. OVERVIEW

2.1 Prerequisites

In this paper we focus on a single autonomous system (AS) since most decisions made by operators or routers are confined to this environment. We presume that routers use an AQM algorithm [10, 4, 8] and that explicit congestion marking is employed by using ECN [20]. AQM algorithms are widely deployed in routers, while ECN is the standard for congestion marking in IP networks. We consider that routers have the ability to compute the paths between any source and destination pair in the network. This is usually the case with link-state protocols. Link-state routing protocols are often used today [18]. We consider that TCP receivers separately acknowledge every data packet received. We defer the discussion of this assumption to Section 3.2.4. Our solution uses only traffic local to the AS since inter-domain traffic can contain congestion markings from other networks.

2.2 Background

An AQM enabled router can be augmented with the ability to mark data packets instead of dropping them. Specifically, the AQM algorithm [10, 4, 8] at a router computes a congestion measure for the router's outgoing links. This measure varies from one algorithm to another. It can be as simple as a function of the size of the router queue (e.g. RED [10]) or a more complex expression based on incoming traffic rate and available bandwidth (e.g. REM [4]). The router then marks each outgoing data packet probabilistically, as a function of the congestion measure of the link they are sent on.

ECN [20] is the protocol that enables congestion marking. It makes use of four bits in the packet headers: the ECN-Echo (ECE) and Congestion Window Reduced (CWR) bits in the TCP header and the ECT (ECN-Capable Transport) and the CE (Congestion Experienced) bits in the IP header. ECN capable packets have the ECT bit set. These packets can be marked by congested routers by setting the CE bit. When a TCP destination receives a packet with the CE bit set it sets the ECE bit in the subsequent ACK and continues to do so for all following ACKs until it receives a data packet with the CWR bit set. The CWR bit is set by a TCP source to signal a reduction in the size of the congestion window. This can happen as a result of receiving an ACK with the ECE bit set or for other reasons.

2.3 Definitions

We use the terms data path or forward path to refer to the path taken by data packets and ACK path or reverse path to refer to the path taken by ACK packets. We call a packet with the ECE bit set a *marked ACK packet* and a packet with the CE bit set a *marked data packet*. As explained, a TCP receiver can mark multiple ACK packets as a result of receiving one marked data packet. Consequently, the sequence and percentage of data packet markings can be modified when the TCP receiver echoes the markings to the ACK packets. We refer to this process as the *alteration* caused by the TCP receiver. While a TCP connection is typically bidirectional, the ECN markings on the two halves of the connection are independent because each TCP source can potentially send data on different paths with different congestion severities. Throughout this paper we also consider a TCP connection as being composed of two standalone halves.

2.4 Overview of the Solution

Our solution allows a router to locally infer the congestion severity of other routers in the network. In this paper we use the marking probability as the representation of congestion severity. A router that has calculated a link level marking probability could obtain the corresponding AQM congestion measure. However, that congestion measure is specific to a particular AQM and its configuration whereas the marking probability is a representation of congestion severity common to several AQM algorithms.

In our solution, routers first analyze the packet markings

in the data and the ACK packets that they receive. The analysis of data packet markings leverages the percentage of marked data packets. The analysis of ACK markings is based on the average size of the sequences of unmarked ACKs. The data packet markings reveal the congestion encountered by the data packet on the path taken until the moment it is analyzed. The markings in the ACK packets reveal the congestion encountered by the corresponding data packets on the entire forward path. From markings that reveal congestion about the same network path a router can infer the aggregate marking probability on that path. After computing path level marking probabilities, a router can leverage hop level path descriptions derived from link-state routing information. Link state routing information is reliably disseminated into the network and is sufficient to allow each router to compute the exact path that a packet takes from a source to a destination. With both path level aggregate marking probabilities and a hop level description of the paths, aggregate marking probabilities for paths that are increasingly shorter can be derived. Thus, link level marking probabilities can also be computed with this approach. A congestion map is composed of the union of all the marking probabilities inferred by a router.

3. INFERRING PATH LEVEL CONGESTION

In this section we describe the first step towards obtaining a congestion map of the network: the inference of path level marking probabilities. Routers can obtain path level marking probabilities from the analysis of packets describing congestion on a network path. Both the markings in the data and the ACK packets are useful for inference. However, the analysis of these two types of markings poses different challenges. The data packet markings are exactly the markings set by routers. The challenge consists in designing a method of aggregating the markings such that accurate and unbiased estimates are obtained. The markings in the ACK packets are not the markings set by the routers. They are the result of the echoing performed by the TCP receiver. The challenge is to design a method for inferring congestion estimates despite the alteration caused by the echoing protocol. We start by tackling the more manageable problem of using data packets for inference.

3.1 Inference Using Data Packets

To take advantage of the congestion markings in data packets a router needs to first choose an interval of time over which a congestion estimate is computed. The length of this interval can vary depending on the final use of the estimate. We call this interval over which a marking probability *estimate* is measured an *estimation interval*. We refer to it as the *EI*. Only one estimate is computed for an EI and it represents the average marking probability for a path for the whole interval. Obtaining the estimates is not as straightforward as computing the percentage of marked data packets received during the EI. This is inadequate because packets are not necessarily received uniformly. Brief periods with plenty or no packets received are always possible. One cause for these variations is the burstiness inherent in the use of TCP. Therefore, the percentage of marked data packets received during an EI is biased towards periods where bursts of data packets are received. This bias is more pronounced as the length of the EI increases. However, we desire the estimate to convey an unbiased view of the marking probability. Ideally, a uniform sampling of the marking probability during an EI needs to be performed. For this purpose we introduce a parameter called the *sampling interval*. For brevity we call it the *SI*. All markings received during an SI are aggregated to form one *sample*. Figure 1 shows the relationship between the EI and the SI. An SI is a subdivision of an EI. The use of the SI can limit the bias described because every estimate is computed by averaging over a number of periodic samples equal to the ratio between the EI and the SI. The value of the SI should be chosen based on the bandwidth in the network. A larger SI may be required for a low bandwidth network in order to obtain a number of packet markings comparable to a high bandwidth network.

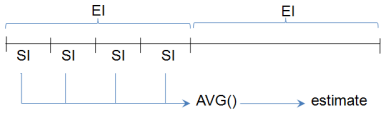

Figure 1: Sampling and estimation intervals

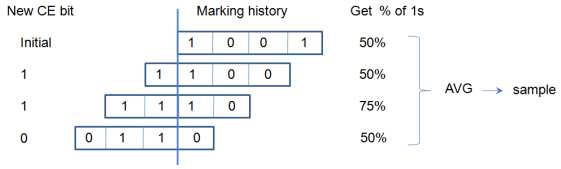

Figure 2: The use of marking history

The SI can, however, artificially separate a group of data packet markings that describe the congestion on a path at the same moment. To ensure that the meaning of the markings is preserved, a data structure called the *marking history* is used. Each router maintains a marking history for each path it measures the marking probability on. The marking history is a sliding bit vector that consists of the last markings encountered in the data packets traversing the same network path. Figure 2 shows a graphical representation of the use of the marking history during an SI. Three data packets markings are received by a router. With every incoming packet, the marking history is shifted one position to the right and the marking from a new packet is added to the tail. After each addition, the percentage of the bits set to 1 is computed. These percentages are finally all averaged to obtain the sample. In a favorable case, when sufficient samples are received every SI, using the marking history or directly computing the percentage of marked data packets are expected to perform similarly. When none or very few markings are received dur-

ing an SI, the sample may not be inferred accurately. However, the router does know that not enough markings were received. It can therefore signal that the inferred estimate needs to be considered with caution.

3.2 Inference Using ACK Packets

In this section, we first demonstrate that simply using the percentage of marked ACKs is inadequate. We then propose two solutions: a basic solution applicable to environments with low or moderate congestion and an improved solution that can also be applied to severely congested environments. We continue to use the concepts of estimation interval (EI) and sampling interval (SI). The difference between ACK and data packet analysis consists in the method used to compute the sample. Because the markings in the ACK packets are an altered version of the markings in the corresponding data packets, a method is needed to account for the alteration. Recall that the alteration consists in multiple ACK packets potentially being marked by the TCP receiver as a result of just one marked data packet.

Reference solution: To understand and evaluate any loss in accuracy when dealing with the alteration, we propose to compare against a reference solution that does not suffer from alteration. This reference solution deals with the alteration by removing its cause. It replaces the ECN echoing algorithm with an alternative echoing scheme in which TCP receivers mark an ACK if and only if the corresponding data packet was marked. As a result, the inference method for data packet analysis can now be extended to ACKs.

While the reference solution solves the alteration problem recall that our goal is an approach backward compatible with the ECN protocol and the end hosts. A natural attempt is to leverage the percentage of marked ACKs similarly to the data packet analysis. We next demonstrate that such a solution is not adequate using a theoretical model that quantifies the effect of the alteration on the inference accuracy. We support our theoretical findings with experimental results. Our solutions for ACK packet inference will become clear from the understanding of the theoretical model.

3.2.1 The Percentage of Marked ACKs is Insufficient

Using the percentage of marked ACKs may result in an overestimate of the marking probability. We next characterize the severity of this overestimation. The severity depends on the congestion window of the flows and the real marking probability. The larger the window, the greater the number of ACKs marked due to a data packet being marked and thus, the bigger the overestimate. However, the bigger the real marking probability, the lower the overestimate since more data packets are already marked. The focus of our theoretical model is to compute a function that maps the inferred marking probability to the window size and the real marking probability when using the percentage of marked ACKs.

To start, let *w* be the window size (e.g. *w* packets are sent every RTT), let *p* be the real marking probability and *I* be the inferred marking probability. For illustration purposes suppose that *w* and *p* are fixed and that a large number of packets are used for inference. Since we assume *w* to be fixed, for each flow there will always be *w* consecutively marked ACKs. These groups of marked ACKs are separated by zero or more unmarked ACKs. Computing *I* as the ratio of marked ACKs will result in the formula:

$$I = \frac{w}{w+q} \tag{1}$$

where *q* is the average size of the groups of unmarked ACKs present between groups of marked ACKs. For computing *q* we use the following equation:

$$q = \sum_{n=0}^{\infty} n * (1-p)^n * p = \frac{1-p}{p} \tag{2}$$

This equation uses the probability of groups of *n* unmarked ACKs to appear in the flow, for any positive value of *n*. A simple substitution in (1) yields the final formula that characterizes the overestimation:

$$I = \frac{w}{w + \frac{1-p}{p}} \tag{3}$$

We design an experiment to test the validity of the theoretical model. We modify the TCP sources in the ns-2 simulator to use a constant window size. For illustration purposes we use a network consisting of one link. For each different experiment we vary the real marking probability on the link by changing the number of TCP flows started. For each single experiment the marking probability is relatively stable because the constant window size sources are randomly started. For computing the samples we use the method described in Section 3.1. We set SI to 0.2s and EI to 30s.

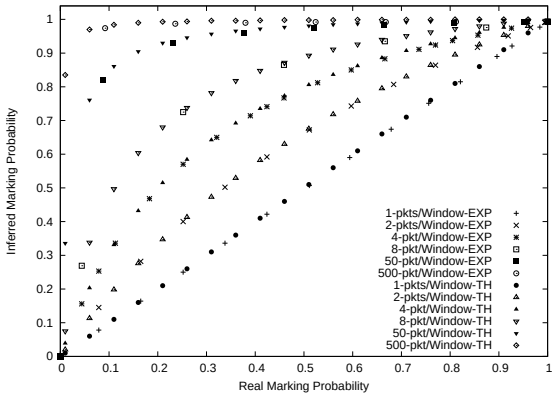

Figure 3: Inferred marking probability vs. real marking probability and window size

Figure 3 plots the inference results of both the theoretical (TH) and experimental (EXP) methods. The experimental results support our theoretical findings. Note the severity of the overestimation. For a window of 8 packets and a real marking probability of 0.15 the inferred marking probability

overestimates by a factor of 4. Therefore, using the percentage of marked ACK packets is not a satisfactory solution.

3.2.2 The Basic Solution

Using equations from the theoretical model, we develop the basic solution for inference using ACK markings. (2) provides a way to compute the real marking probability *p*. Recall that *q* is the average size of the groups of unmarked ACKs. Note, however, that the equation includes the probabilities for groups of unmarked ACKs of any size, including size zero. Such groups appear when the CWR packet that signals the TCP receiver to stop setting the ECE bit is also marked. The basic solution does not use the groups of size zero. Let q_{basic} be the average size of all groups of unmarked ACKs larger than zero. Removing the groups of size zero reduces the fraction of the groups useful to the basic solution to $1-p$ but does not change the number of unmarked ACKs. Therefore, as (4) shows, q_{basic} simply becomes the inverse of *p*. We use *U* to denote the number of unmarked ACKs and $G_{>0}$ and $G_{\geq 0}$ to denote the number of groups of unmarked ACKs larger than zero or including size zero. If q_{basic} or U is 0 there is not enough information to compute a sample. Routers can choose to fall back to using the previous sample or consider the marking probability to be zero.

$$q_{basic} = \frac{U}{G_{>0}} = \frac{U}{(G_{\geq 0})(1-p)} = \frac{q}{1-p} = \frac{1}{p} \quad (4)$$

To leverage (4), routers monitor a number of flows during an EI. The process is depicted in Figure 4. The groups of unmarked ACKs and their sizes are counted. Finally, q_{basic} can be computed and *p* can be trivially derived. Note that the state required for the computation is small and comprised only of simple counters. Moreover, we later show that only a small number of flows need to be monitored.

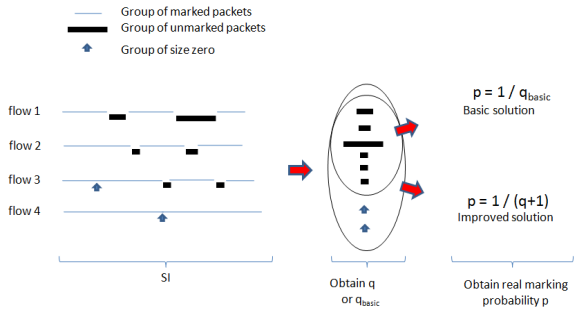

Figure 4: The computation of a sample

Each EI is treated separately. Therefore, the set of monitored flows from different EIs may not bear resemblance. Flows are used for inference starting with the SI following the SI where they are first encountered. To ensure that there are flows usable in the first SI of an EI, routers first have a training period at the beginning of each new EI. During this period flows are monitored as in a regular SI, however, this period does not count towards computing of the estimate.

Groups of unmarked ACKs can span multiple SIs. They are counted in the SI where they end. Incomplete groups of unmarked ACKs may be encountered at the start or end of an EI. Such groups begin or end in a different EI. Since each EI is treated separately, the correct size of incomplete groups cannot be correctly measured and this can skew the results. Thus, incomplete groups are not considered for inference.

The probability for a group of unmarked ACKs to be of any size greater than zero is $1-p$. Therefore, in scenarios where *p* is not close to 1, the basic solution can be applied because a large number of groups can be available.

3.2.3 The Improved Solution

In environments prone to high levels of congestion the number of groups of unmarked ACKs of size greater than zero may not be enough to provide statistical significance. In such scenarios a more accurate solution is desired. If groups of size zero can also be accounted for, the inference can make use of a significantly larger number of groups. This is because groups of size zero appear with probability *p* and therefore become predominant when *p* is large.

Nevertheless, counting groups of size zero is not trivial. A group of size zero can only be identified by tracking the CWR packets and identifying their corresponding ACKs. Consequently, only routers that have access to both the data and the ACK path of a flow can use this improvement. This however, does not require routing to be symmetric. If the first-hop and last-hop router for a flow are unique, as is usually the case, those routers will always be able to analyze both data and ACK packets, irrespective of the degree of asymmetry of the entire end-to-end path.

In our improved solution, a router remembers the sequence numbers of the last byte of the CWR packets it observes in the monitored flows. In the common case, only one CWR packet per RTT is expected for a given flow and it can be discarded after it is matched to the corresponding ACK. The ACK corresponding to a CWR packet can be identified by its sequence number which is the first value larger than the remembered value. If both the ACK corresponding to a CWR packet and the previous ACK are marked, this signals the presence of a group of unmarked ACKs of size zero. To ensure that every ACK packet can be checked against its corresponding data packet, for each flow, the sequence number of the first detected data packet is stored and only the ACKs with a greater sequence number are considered.

In a network with asymmetric routing, our improved solution trades the number of routers that can use the method for an improvement in accuracy. This improvement, makes it suitable for a larger number of congestion environments than the basic solution.

3.2.4 Effects of Delayed ACK

So far we considered a TCP receiver separately acknowledges every packet. The delayed ACK algorithm [6] allows a TCP receiver to briefly delay the acknowledgment of a re-

ceived data packet so that up to two data packets can be acknowledged at once. The congestion marking echoed in the ACK packet becomes the logical disjunction of the markings in the two data packets [20]. This process alters the number and sizes of the sequences of unmarked ACKs compared to a TCP that does not use delayed ACK. We next perform a numerical analysis on the effect of delayed ACK on our improved solution. We consider a very large number of data packet markings set with some marking probability. An ACK marking is created for every two data packets according to the delayed ACK algorithm. Because many markings are used, the improved solution can correctly infer the marking probability. This allows us to quantify the *additional* inaccuracy caused by using delayed ACK. The results are shown in Figure 5. Note that even though the delayed ACKs cause an overestimation in the inference, the severity of the inferred marking probabilities for both cases is comparable.

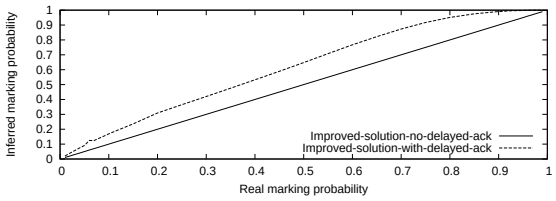

Figure 5: Numerical analysis on effect of delayed ACK

4. CONSTRUCTING THE CONGESTION MAP

In this section, we describe the building blocks used for computing a detailed congestion map of the network and then discuss the extent of the map's coverage. We also present potential challenges in computing the congestion map. Recall that the congestion map is composed of all marking probabilities inferred by a router.

4.1 Building Blocks

The first building block is the set of path level marking probabilities that a router can infer. These marking probabilities can then be further broken down using the second building block; a set of equations that directly result from leveraging link state routing information.

For the sake of clarity, in this discussion we use the simplifying assumption that the marking probability on a link is constant. However, in practice this assumption is not necessary. We use the sample network in Figure 6 to support our explanations. Bidirectional links connect the routers. Routing is shortest-path and each link is labeled with its weight. Unless otherwise stated, the route through R_5 is not used. We use S_{ij} to denote the entire path that takes traffic from some router R_i to some router R_j and P_{ij} to denote the estimate of the marking probability over S_{ij}. If R_i and R_j are neighbors, $P_{ij} = L_{ij}$, where L_{ij} is the link level estimate.

The first building block is the set of path level marking probabilities that a router can infer only from the analysis of packet markings. The extent of this set depends on the positioning of the router with respect to the traffic it monitors, which traffic is monitored, and whether the basic or improved solution is used. We characterize this extent by discussing all possible scenarios that are useful in practice. Each scenario is explicitly labeled with which solution it is applicable to and which traffic is monitored.

Scenario 1 - Only data markings: A router R_i that observes only the data packets from the traffic sent by some source R_s can infer P_{si}. In other words, R_i will be able to infer the marking probability on all the paths that carry data traffic to it. In Figure 6, if R_0 and R_1 send TCP traffic to R_4, then, R_2 will be able to infer P_{02} and $P_{12} = L_{12}$.

Scenario 2 - Only ACK markings - Using the basic solution: A router R_i that observes only the ACK packets from the traffic between a receiver R_d and a source R_s can infer P_{sd}. In other words, R_i can infer the marking probability over the entire forward path from R_s to R_d. In Figure 6, consider that R_0 sends traffic to R_4 using R_1 as a next hop but the reverse ACK traffic goes through R_5. In this case, from the analysis of the ACK packets, R_5 infers P_{04}.

Scenario 3 - ACK and data markings - Using the basic or improved solution: A router R_i that observes both the ACK and data packets sent between some sender R_s and some receiver R_d can infer both P_{si} and P_{sd}. P_{si} can be computed as described in the first scenario. P_{sd} can be computed by the basic solution as in the second scenario and by the improved solution by matching the information from both ACK and data packets. In our example suppose there is traffic from R_0 to R_4 and it is taking the route through R_1. R_1 can infer P_{01} by analyzing the data packets and P_{04} with the help of the ACK packets.

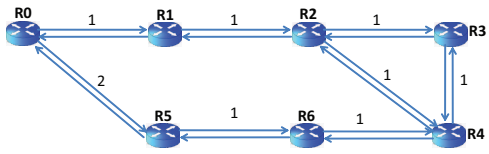

Figure 6: A sample network

All scenarios described above regularly appear in practice. Let Fwd_{sd} and Rev_{sd} be the set of routers on the forward path and the reverse path for traffic generated by R_s for R_d. If $R_i \in Fwd_{sd} \bigcap Rev_{sd}$ then Scenario 3 applies. If routing is asymmetric, then, a router can possibly find itself on either a data path or an ACK path for which it is neither source nor destination. If $R_i \in Fwd_{sd} \setminus Rev_{sd}$ Scenario 1 applies. If $R_i \in Rev_{sd} \setminus Fwd_{sd}$ Scenario 2 applies.

The second building block for computing a detailed congestion map of the network is the method used for breaking down the inferred aggregate path level marking probabilities. Link-state routing information plays a vital role in this process. Using link-state information, a router can compute the set of links that are part of the paths for which the marking probability has been inferred. This knowledge can help

to further break down marking probabilities to describe increasingly shorter portions of a path. We can formally represent these ideas as follows.

A router R_i can compute the link or path level estimate P_{jk} if it knows P_{tj} and P_{tk} ($t = i$ usually, but not necessarily), and S_{tj} is a strict subset of S_{tk}:

$$P_{tj} + (1 - P_{tj})P_{jk} = P_{tk} \rightarrow P_{jk} = \frac{P_{tk} - P_{tj}}{1 - P_{tj}} \quad (5)$$

A similar equation can be derived if P_{jt} and P_{kt} are known, and S_{kt} is a strict subset of S_{jt}:

$$P_{jk} = \frac{P_{jt} - P_{kt}}{1 - P_{kt}} \quad (6)$$

To exemplify the use of the formulas, consider the third scenario described, where P_{si} and P_{sd} are inferred. Using (5) a router can compute P_{id}. For example, from P_{01} and P_{04} a router can derive P_{14}. In other words, in the third scenario, R_i can now obtain the marking probability on both the paths carrying traffic to it and away from it.

In (5) we considered that if S_{tj} is a strict subset of S_{tk} then $P_{tj} \leq P_{tk}$. However, this inequality may not always hold in practice. For example, if S_{jk} is uncongested then the values of P_{tj} and P_{tk} should be equal. In practice, small differences will appear. Possible reasons are routers choosing different flows to monitor or considering the same markings as part of different SIs. In all these cases, if the quantity $P_{tk} - P_{tj}$ is negative then zero should be used instead. The same argument applies to (6).

4.2 Extent of Congestion Map Coverage

All-to-all traffic pattern: When traffic is flowing between every pair of routers in the network (all-to-all traffic pattern) a router R_i's congestion map contains estimates for all links that carry data traffic to and away from R_i. Since R_i is the first hop for a number of flows, it will receive both the data and ACK packets from them. Therefore, it can use Scenario 3 to compute an estimate for the paths to all other routers in the network. (5) and (6) can then be used to calculate the link level estimates. The all-to-all traffic pattern is very common today. For example, a large number of networks deploy routers to aggregate traffic from entire cities, and there is usually traffic flowing between any two cities.

The limit of the coverage: A congestion map may not always contain all link level marking probabilities. One case is when markings from a link never reach a router that performs the inference. For example, the link (R_3, R_4) never carries packets from flows that reach R_0, since it is not on the shortest path from R_0 to any other router. Therefore, R_0 will not be able to infer $P_{34} = L_{34}$. To analyze what percentage of the link-level marking probabilities can be inferred in practice, we analyze the following real network topologies: Internet2, TEIN2 (Trans-Eurasia Network), iLight (Indiana's Optical Network), GEANT (European research network), SUNET (Sweden) and NLR (National LambdaRail). We find that if an all-to-all traffic pattern exists, on average, the congestion map of a router from NLR, Internet2 and GEANT contains around 60% of the link level marking probabilities. However, since the remaining links cannot carry traffic to the inferring routers under the current routing protocol configuration, they will likely be less useful to those routers. If the links begin to carry traffic, their congestion level can be quickly inferred. Moreover, the maps of the routers from TEIN2, iLight and SUNET contain on average 91%, 94% and 95% of the link level probabilities. In these cases, a hybrid approach that also leverages polling can fill the missing link level marking probabilities because the percentage of links that require polling is very small.

A second case where maps may not contain all link level marking probabilities concerns traffic patterns that are not all-to-all. In these cases some path level marking probabilities cannot be completely broken down into link level probabilities. This can happen, for example, when one of the routers on some path is not the first hop for any traffic that reaches R_i. In our example, suppose R_0 only sends traffic to R_5 and R_4. It only uses the path through R_5 because link (R_1, R_2) failed. While the link (R_6, R_4) does carry traffic to and from R_0, R_0 will only be able to infer P_{54} and not P_{64}. This is because R_0 does not receive any packets that have R_6 as a first hop. Nevertheless, the path level marking probabilities also provide useful information to applications.

4.3 Path Level Visibility

One challenge in computing a congestion map of a network is the limit to the amount of information that the packet markings can encode. For example, consider the case when a router R_i computes P_{ij}, the aggregate marking probability to R_j. Let P_{ij} be very close to 1 and suppose packets to the subsequent hop R_{j+1} take the same route to R_j. Because most packets are already marked at R_j it is difficult to encode additional congestion information for the link (R_j, R_{j+1}). We call this "limited visibility". In the extreme case, when P_{ij} is 1 no new congestion information can be encoded in the markings.

To provide more insight into the effects of the "limited visibility" we perform an experiment. We use the modified TCP sources that send traffic using a fixed congestion window. The choice of a fixed TCP window decouples visibility from the effects of TCP's congestion control algorithm because flows no longer get throttled down. Moreover, the fixed TCP window sources allow the marking probabilities to remain nearly constant and this allows us to better track the visibility problem as a function of the marking probability. In this experiment we use a TCP window of 2. The exact value is inconsequential but a small value allows us to better control the increments in the marking probability.

We use a simple chain network consisting of three links, from router R_0 to R_3. Flows are started from R_0 to all other routers, from R_1 to R_2 and from R_2 to R_3. We fix the marking probability on the link (R_2, R_3) at 0.36 (any value is

equally good) by starting a fixed number of TCP flows from R_2 to R_3. We vary the aggregate marking probability on the link (R_1, R_2) by starting different numbers of TCP flows between R_1 and R_2. The purpose of the experiment is to see the degradation in the inference for the link (R_2, R_3) when the marking probability on (R_1, R_2) increases to 1.

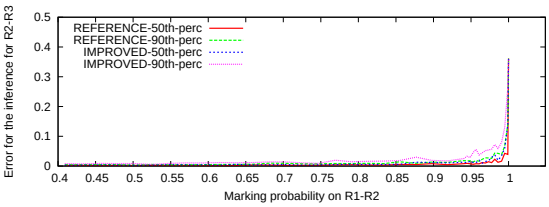

Figure 7: The effect of limited visibility

The results are shown in Figure 7. We present the 50th and 90th percentiles of the absolute difference between the real and the inferred marking probability. Even though a large number of packets are available for analysis, the accuracy of the estimation drops significantly as the marking probability on link (R_1, R_2) increases to 1. As fewer and fewer packets escape unmarked from router R_1, it is increasingly more difficult to encode the marking probability on (R_2, R_3). In the extreme case (marking probability of 1) there are no more packets unmarked that can carry useful information about (R_2, R_3) and therefore, R_0 considers the marking probability to be 0. The resulting error is exactly 0.36, the value of the marking probability on (R_2, R_3). An exact point at which the visibility effect begins to appear cannot be precisely determined as it depends on the number of packets received, the aggregate marking probability on the paths and the variation of the marking probability on the path we want to measure ((R_2, R_3) in our example). In conclusion, the "limited visibility" can appear irrespective of the number of packets used for analysis. However, a significant level of congestion is necessary to limit the visibility. In our experiment the inference accuracy is good even when more than 95% of the packets are marked.

5. EVALUATION

5.1 Methodology

We conduct simulations to evaluate our solutions because they allow us to freely and severely congest network environments. For this purpose we augmented the ns-2 simulator with our inference methods. We next describe our default experimental setup. Exceptions will be discussed separately, together with the experiments they apply to.

Network environment: The function used by AQM algorithms to map the congestion measure to the marking probability influences the inference accuracy. We evaluate the effects of both the linear functions (PI [8], RED[10]) and exponential functions (REM [4]) present in ns-2 AQMs. By default, we use a linear marking function. We use RED to represent this group of functions since it is standardized and present in many routers [7, 14]. For RED we disable the waiting between marked packets (the ns-2 wait_ parameter) in order to be compliant with the RFC. We set the marking probability to linearly increase to 1 (the ns-2 max_p_ parameter) as the average queue size grows to max_thresh. Both the queue measuring function and the dropping probability are defined per-byte. We set min_thresh and max_thresh to 25% and 75% of the buffer size. Router buffer size is equal to the product between the link capacity and the average round-trip time in the network as is commonly the case today [3].

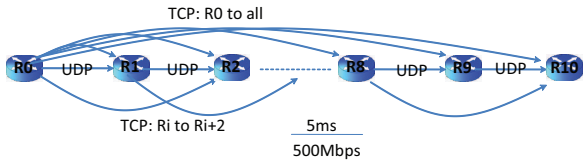

Figure 8: Topology and traffic pattern

Why a chain topology suffices: The setup for the topology used is described in Figure 8. It is a chain topology with 10 links. We deliberately chose this type of topology because the computation performed by our inference algorithm on a chain topology is a generalization of the computation for any conceivable topology. The reason is that for any topology the inference algorithm is based on path level computations and each path is essentially a chain topology. Link bandwidth is limited to 500Mbps in order to keep the simulation times tractable. However, we also discuss the effects of changes in bandwidth. The propagation delay on each link in the topology is 5ms. We use node R_0 as a vantage point. All the inferences presented are from the point of view of node R_0.

Traffic: We use TCP Reno and UDP sources to generate traffic. Packet size is 500 bytes. The TCP flows used are FTP flows that once started last for the entire duration of the simulation. However, in subsection 5.4.5 we also evaluate the influence of small flows. Our TCP sources separately acknowledge every data packet because we cannot control the extent to which a delayed ACK functionality would be used. There is TCP traffic from node R_0 to each of the other nodes, and the number of such flows is computed using the expression:

$$Nr.\ Flows\ from\ R_0\ to\ R_i\ =\ N * i^2 \qquad (7)$$

The parameter N can be tuned and the default value is 250. Note that the number of flows started from R_0 to the other routers increases with the distance from R_0. We made this choice in order to minimize the bias that TCP has against flows with larger RTTs. Background traffic is simulated by TCP traffic from any node R_i to node R_{i+2}. A fixed number of 100 flows are started between such pairs of routers. We use the UDP sources as a method of inducing variation in the per-EI congestion level in the network. We devised a custom UDP source that changes its sending rate by a per-

centage of the link bandwidth every second while continuously cycling between 0 and 500Mpbs. The default value is 2% (10Mpbs). Such sources are started between every consecutive pair of routers. Note that the use of UDP sources reduces the number of TCP markings used by the inference. In reality, networks may see a significantly smaller percentage of UDP traffic. In those cases the accuracy of our solutions could improve. All the TCP and UDP flows described are set up on the forward path. After the flows are started all ten links become permanently congested and show decreasing marking probabilities ranging from roughly 0.3 for (R_0, R_1) to 0.12 for (R_8, R_9) for each EI. The link (R_9, R_{10}) marks packets with roughly 0.05 probability.

Metric and solution parameter values: Unless otherwise stated an SI of 0.5s and EI of 3s are used. Each simulation runs for 500s. The results include the initial phase in which flows are started are congestion suddenly ramps up. The packet history for the reference solution is fixed at 10 packets. The number of monitored flows is capped at 1000. Note that this is the maximum allowed; some EIs will observe less flows. The training period for the improved solution is 2 SIs (1 second).

To quantify the inference accuracy we use the 50th and the 90th percentile of the absolute difference between the inferred and real link level marking probabilities.

$$Accuracy\ Metric = |Inferred\ Mark.\ Prob. - Real\ Mark.\ Prob.| \quad (8)$$

Let EI_s and EI_f be the start and finish time of an EI used by R_0 for computing the inferred marking probability for some link (R_i, R_j). The estimate cannot be directly compared to the real marking probability on the link from time EI_s to EI_f because it takes time for the markings to travel from (R_i, R_j) to R_0. For a fair comparison, to compute the real marking probability on (R_i, R_j), we consider P_a and P_z, the first and last packets received by R_0 from R_j during the EI. Let EI_a and EI_z be the times at which each of these packets are received by R_0. $EI_s \leq EI_a \leq EI_z \leq EI_f$. Also let t_a and t_z be the times at which these packets were marked by R_j. The real marking probability against which we compare is the weighted average over all marking probabilities at R_j over the interval that starts at $t_a - (EI_a - EI_s)$ and ends at $t_z + (EI_f - EI_z)$. If no packets are received from R_j during an EI the real marking probability is averaged over the EI.

We perform numerical comparisons between the inferred and real marking probabilities. A reasonable approach would have been to quantify congestion based on the severity (e.g. low, medium). In practice, most applications should be content with such a discrete representation of congestion. However, performing direct numerical comparisons allows us to better present the strengths and limitations of our approach.

Description of the experiments: For evaluation we focus on the analysis of downstream links using ACK markings since the alteration at the receiver makes the inference challenging and more susceptible to inaccuracies. In comparison, the inference of upstream links using data packets is relatively trivial. Nonetheless, the accuracy of the data marking inference is similar to that of our reference solution since neither of them suffers from the alteration at the receiver. In all experiments care was taken to minimize the possibility of the visibility problem appearing by limiting the aggregate marking probabilities to roughly 0.85. We first compare the basic and improved solution. We then use the improved solution for the rest of the experiments to evaluate the effect of parameter values and various network conditions.

5.2 Basic vs Improved Solution

5.2.1 Low-Medium Aggregate Marking Probability

We first discuss a scenario with a single bottleneck link as this is often encountered in practice on the ingress/egress links of access networks. We halve the bandwidth of the link (R_9, R_{10}) to create a bottleneck. We then start additional flows from R_9 to R_{10} to obtain different congestion levels. As a result of the traffic pattern, link (R_0, R_1) also becomes congested. All other links remain uncongested. The results of the inference for (R_9, R_{10}) are shown in Figure 9.

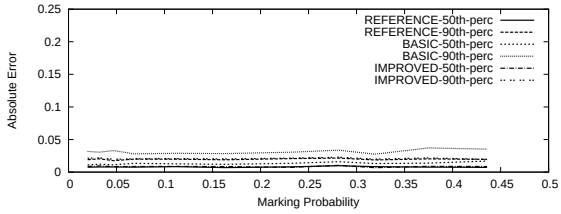

Figure 9: Bottleneck link with low to medium aggregate marking probability

For low or medium marking probabilities on the bottlenecked link the accuracy of the inference is very high for all solutions. The error is at most 0.04. Note that even very small marking probabilities are inferred with very good accuracy. Inferring small marking probabilities is more difficult since large groups of unmarked packets are more likely to appear and may be larger than an SI. Notice also that the inference error does not increase with the marking probability because in all cases the solutions obtain enough data points for analysis. The basic solution performs only slightly worse and this is because it does not make use of the additional data points offered by the groups of size zero.

5.2.2 High Marking Probabilities

While networks with single bottlenecks are common, we wish to test our solution on complex scenarios with multiple congestion points. Such scenarios are more likely to exhibit increased aggregate marking probabilities. In these scenarios the groups of size zero become predominant and the inference accuracy for the basic solution, therefore, decreases. In Figure 10 we plot the inference error for the three solutions for our standard scenario for an EI of 3s. The ag-

gregate marking probability at the second hop is already 0.48 and increases to nearly 0.84 for the last hop. In this case the accuracy of the basic solution decreases faster at higher aggregate marking probabilities because it does not leverage the groups of size zero. Nevertheless, the results are good for the first few hops, when the aggregate marking probabilities are not very high. In practice, in the common case, we also do not expect very high marking probabilities.

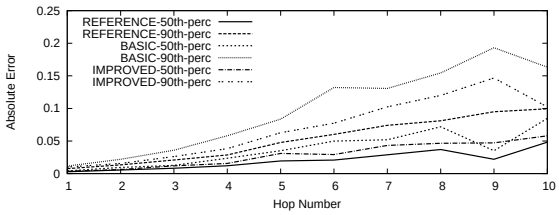

Figure 10: Effect of high aggregate marking probability

5.3 Sensitivity to Parameter Values

In the following experiments we use environments with multiple congestion points and high aggregate marking probabilities because they are more challenging. We focus only on the accuracy of the improved solution since in these scenarios it outperforms the basic solution.

5.3.1 Sensitivity to the Length of the EI

In this experiment we analyze how the length of the EI affects the accuracy of the results. For this we fix the SI at 0.5s (roughly the average RTT of our topology) and vary the ratio between the EI and the SI. The results are shown in Figure 11. The x-axis is in logarithmic scale.

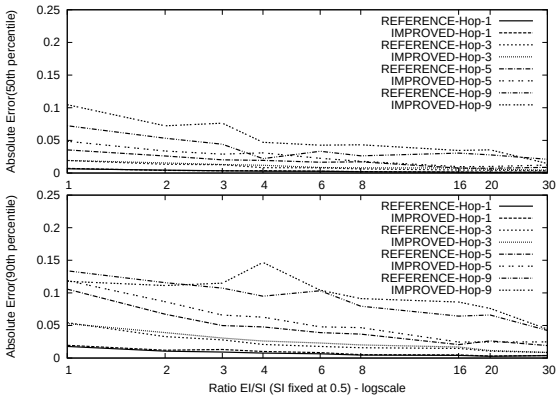

Figure 11: Error vs ratio of EI and SI

Note that the results of the improved solution are very close to the reference solution. The inference for small EI/SI ratios is visibly more error-prone than for larger ratios. For small ratios fewer packets and groups are available for inference. This is especially important for distant hops as the RTT for those flows is larger. Therefore, the accuracy of the inference decreases with an increase in hop count. Another reason why the accuracy decreases with the hop count is that flows traverse more congested links. Each congested link introduces small variations in the aggregate marking probability as flows cannot all be marked with precisely the same probability. Nevertheless, even in such a network where every link is congested, for an EI of 3s (ratio 6) the 90th percentile of the error for the 5th hop is within 0.05. If an EI of 15s is used, the error for all hops is within 0.05.

5.3.2 Sensitivity to Number of Monitored Flows

In this experiment we vary the number of monitored flows for a path and analyze the impact on the estimation accuracy. Intuitively, the more flows monitored the better the accuracy because more data points are available for the inference. The results are shown in Table 1. Only hop 6 is represented because the results for all other hops show the same trend. The numbers for the packets received and flows monitored are averaged over all the EIs in an experiment.

Flows Monitored	Groups/SI Hop 6	Improved Sol Hop 6 (50th)	Improved Sol Hop 6 (90th)
100	32	0.031	0.152
250	89	0.042	0.119
500	172	0.035	0.069
742	193	0.038	0.087
1233	267	0.027	0.072
1593	375	0.032	0.067

Table 1: Error vs number of monitored flows

When few flows are monitored the accuracy suffers. This is particularly visible when the 90th percentile is considered. Few flows cannot provide enough data samples. This is especially true during congestion when flows are more likely to decrease their sending rate. More importantly, monitoring a very larger number of flows provides diminishing returns in accuracy. This suggests that monitoring a small, constant number of flows is enough to obtain good accuracy.

5.4 Sensitivity to the Network Environment

5.4.1 False Positives

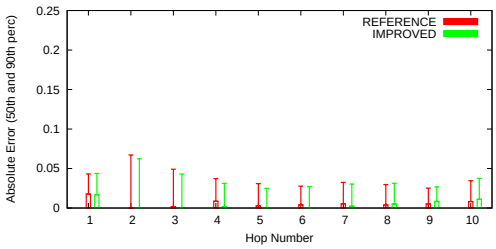

Figure 12: Error on uncongested links vs hop count. Box is 50th percentile, errorbar is 90th percentile

In this experiment we congest only the first link in the network. For this, we start a variable number of TCP flows between routers R_0 and R_1. The rest of the links are kept uncongested by removing all the traffic except if started from

R_0. We vary the percentage with which the UDP sources change their sending rate from 50Mbps to 200Mbps in 25Mbps increments. The results from all experiments are very similar. For one experiment, we show the inference errors for the uncongested links in Figure 12. The marking probability on the uncongested links is sometimes overestimated. This is inherent in the design of the marking algorithm, since it is a probabilistic approach and there is always the danger of minute differences in the marking probability applied to flows from different hops. Nevertheless, the overestimates are very small. Therefore, if coarse congestion estimates (e.g low, high) are used the severity of the congestion will be correctly inferred in most cases.

5.4.2 *Sensitivity to Different AQM Algorithms*

We next use different AQM algorithms to test our solution. Alongside RED, we also evaluate REM and PI. The parameters used for REM and PI are the default ns-2 values. The inference error will depend on the function that the AQM algorithm uses for mapping congestion measures to marking probabilities. For REM this is an exponential function. It creates abrupt variations in the marking probability for small changes in the congestion measure. RED and PI use a linear function. The results are shown in Table 2 for the improved solution. The inference for the reference solution yields very similar results. As expected, REM does not perform well. The exponential function it uses is far more likely to produce a visibility problem compared to the linear function of RED and PI. In our experiment, the visibility problem appears as soon as the second hop. In fact, it is known that REM's aggressive packet dropping/marking behavior can cause problems for flow [17]. On the other hand PI exhibits good performance, similar to RED.

Hop	RED-50th	RED-90th	PI-50th	PI-90th	REM-50th	REM-90th
1	0.002	0.006	0.026	0.035	0.001	0.014
3	0.009	0.023	0.028	0.068	0.999	0.999
5	0.014	0.034	0.035	0.103	0.999	0.999
7	0.030	0.068	0.047	0.132	0.999	0.999
9	0.036	0.096	0.065	0.164	0.999	0.999
10	0.055	0.086	0.079	0.175	0.999	0.999

Table 2: Absolute error for different AQMs

5.4.3 *Performance in Higher Bandwidth Environments*

This experiment analyzes the sensitivity of the inference to an increase in bandwidth. We start with a link bandwidth of 100Mbps and go up to 1Gbps. Intuitively, an increase in bandwidth provides more groups for the inference.

Results are shown in Figure 13. As bandwidth increases there are more packets exchanged between R_0 and the other routers. The number of groups of unmarked packets available for inference also increases with the bandwidth. Since the inference process benefits from more data points the accuracy increases with the bandwidth. Note that the improvement diminishes when bandwidth is scaled up to 1Gbps because the inference process is already provided with sufficient data points. Note also the increased variability in the inference for the last hop. This is caused by the fewer number of packets and groups received from the last hop.

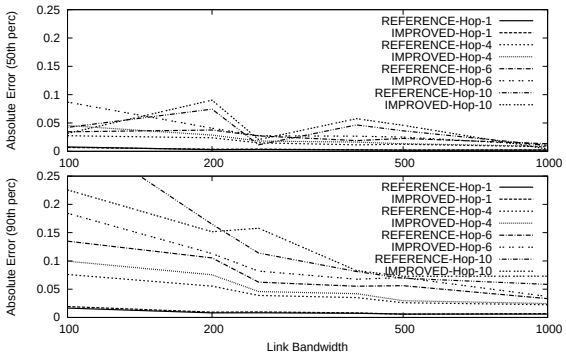

Figure 13: Higher bandwidth environments

5.4.4 *Sensitivity to Transmission Delays*

Although different flows may be marked roughly simultaneously on the forward path, the corresponding ACK packets may not be encountered simultaneously on the reverse path. One factor we have already accounted for is background traffic. Another factor is the variable delay incurred at the TCP receiver even for flows that share the same first hop router. Possible causes are the use of virtualization at the end host or the presence of heavy load. To evaluate the effect of this delay on the inference we first set a maximum delay that can be incurred at the TCP receiver. For each router R_i with which R_0 is exchanging traffic we compute ten random delay levels between zero and the maximum defined delay. Flows received by R_i from R_0 randomly use one of these values to delay the traffic.

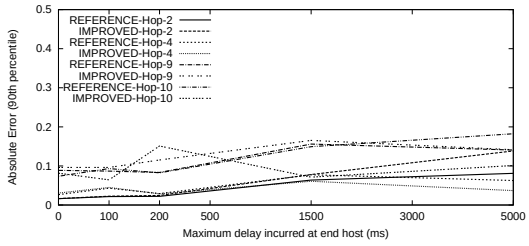

Figure 14: Sensitivity to transmission delays. EI=3s

The results are shown in Figure 14. If the maximum delay at the receiver is under 200ms the effect is minimal because most markings fall in the same EI as for the case with no delay. 200ms is already a large value even for the RTT of most flows [11]. Nevertheless, we also study the effect of larger delays. Once the 1500ms mark is reached, the accuracy decreases. Groups of packets that would fall in the same EI are distributed by the extra delay in different EIs. Also, note the reference solution is similarly affected. This experiment

shows that in the common case we do not expect the delay at the TCP receiver to affect the inference results.

5.4.5 *Sensitivity to Flow Size*

In the previous experiments we used long TCP flows. However, a significant number of flows today are small flows. To simulate such scenarios we consider a percentage of the total number of flows to be small flows. We use 10%, 50% and 90% as different values. We then limit the number of packets counted from these flows. If the limit is 2 then only the first 2 counted packets of a flow are considered for inference. The results are presented in Figure 15.

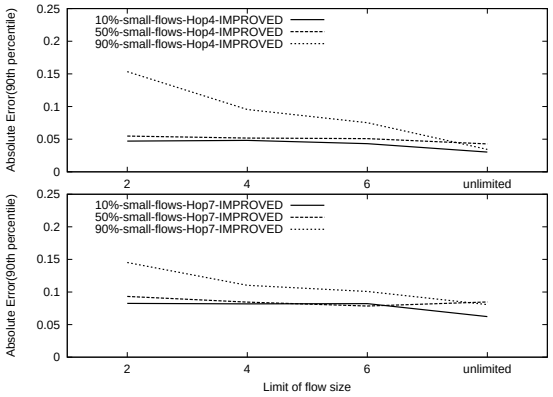

Figure 15: Error vs % of small flows vs limit on flow size

If 90% of the monitored flows are small flows, the effect on the accuracy become visible. However, even when 50% of the flows are small the inference result is good. This is because over all flows there are still enough groups for inference. We also evaluated the reference solution. As expected, the reference solution is less affected by the size of the flows because it uses packets instead of groups of packets. This experiment shows that accurate results can be obtained even when a significant number of flows are small flows.

5.4.6 *Sensitivity to Dropped Packets*

With AQM/ECN networks packet loss should be relatively rare because packet markings are used as early congestion warnings for TCP flows. Nevertheless, in this subsection we analyze the effect of dropped data and ACK packets. For our solutions, in the ideal case, the distribution and sizes of the sequences of unmarked ACKs that a router observes should be consistent with the aggregate marking probability on the corresponding forward path. Similarly, the percentage of marked data packets at any point should be consistent with the aggregate marking probability encountered on the path so far. Dropped ACK and data packets could affect these conditions. Drops may also limit the number of packets available to the inference algorithms.

Dropped data packets: In an AQM/ECN network, data packets are ECN capable and therefore should be rarely dropped. Nevertheless, consider a router that drops data packets with some probability d. One example of such a router is an AQM router not ECN compatible but present on the same path with ECN compatible routers. The probabilistic drop will not change the marking probability. Therefore, the percentage of marked data packets and the sequences of unmarked ACKs will still be consistent with the marking probability. Effects on the inference accuracy may still occur when too few packets are left for inference.

To validate our intuition we look at the effect of data packet loss on the ACK packet inference. We replace some of the routers with non-ECN capable routers that drop packets instead of marking them. We present results for the 6th hop in our topology. We run several experiments where any one or any two of the first five links are not ECN capable. Since flows react similarly to both drops and markings, changing the ECN capability of some routers did not change the level of congestion in the network. Therefore, since in our scenario all routers are congested, the non-ECN capable links actually drop data packets. In the following table we present the worst-case results for both the scenarios where one or two of the first five links are non-ECN capable.

Scenario Worst-Case	Reference Sol Hop 6 (50th)	Reference Sol Hop 6 (90th)	Improved Sol Hop 6 (50th)	Improved Sol Hop 6 (90th)
no link	0.022	0.046	0.022	0.048
1 link	0.020	0.05	0.027	0.062
2 links	0.017	0.051	0.023	0.057

The results confirm our intuition. Dropped data packets have a negligible effect on the inference accuracy of the improved solution as long as enough groups are still available for inference. As expected, the reference solution is also unaffected because the percentage of marked ACKs remains roughly the same despite the dropped packets.

Dropped ACK packets: Not all packets are ECN capable. ACKs piggybacked on data packets are ECN capable, but pure ACKs are not and therefore can be dropped during congestion. Since our solutions use the sequences of ACKs for computation, ACK losses can impact the inference accuracy. In this experiment we analyze the degree of inaccuracy that the loss of ACKs *adds* to our improved solution. We use a numerical model where a large number of markings are created with some probability p and dropped with a probability d for flows with a window size of w. This model allows us to discuss results for all values of d, p and w. This would not be possible in a realistic simulation since we cannot precisely control these parameters. Moreover, because a large number of markings are used, the error of the improved solution is zero. This allow us to precisely quantify the additional degree of inaccuracy that ACK loss induces.

The worst case error appears for large windows. This case is pictured in Figure 16 for w of 50. The inaccuracy appears because the reduction in the number of unmarked ACKs due to the drops is not coupled with a similar reduction in the number of groups of unmarked packets. Please note that d must be in excess of 0.5 for the inference error to top 0.1. Such a high drop probability is never desired in a network as it significantly degrades end-user performance. If w de-

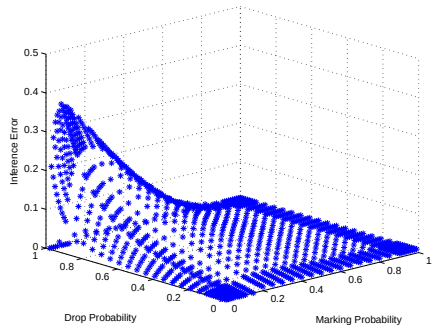

Figure 16: Numerical analysis on the additional inaccuracy caused by dropped ACKs

creases the error also decreases because there is a greater chance of dropping all the markings between groups of unmarked packets. This will cause groups to unite which leads to smaller errors compared to a larger *w*. For *w* of 4 packets *d* must be in excess of 0.65 for the error to top 0.1.

5.4.7 *Sensitivity to Sudden Changes*

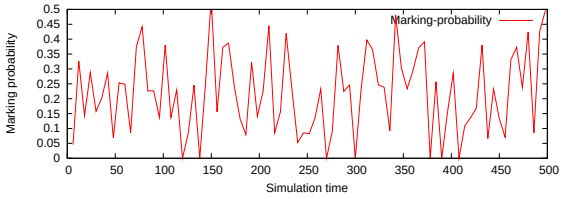

Figure 17: Marking probability on the second hop

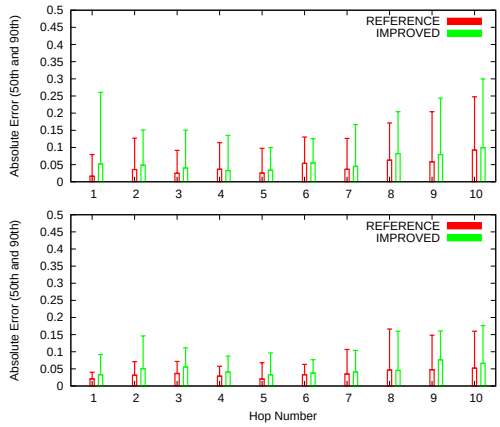

Figure 18: Error for EI of 3 and 10 for each hop

The default behavior of RED is to smooth out variations in the congestion severity. However, we also wish to test our solution in scenarios where the congestion severity and therefore the marking probability varies suddenly and substantially. To introduce these heavy variations we instruct the UDP sources to change their sending rate by 50Mbps (10% of link bandwidth) every second while cycling between 250Mbps and 750Mbps (50%, 150% of link bandwidth). Every 10s we also stop the UDP sources for a random duration between 0s and 10s. Every 10s we start 3000 TCP flows between a random pair of nodes in the network. Each of these 3000 flows finishes after sending for a random time between 0s and 10s. The resulting marking probability variation at the EI scale for the second hop is presented in Figure 17. The other hops present a very similar pattern of variation. Note that packet drops are often encountered and the limited visibility is also a factor, albeit transiently.

The results are presented in Figure 18. Even with these sudden changes in the marking probability and with an EI of just 3s the 50th percentile of the error is still under 0.05 for the first seven hops. The inference for the 1st hop is affected by the small number of flows started to R_1. As a result, very few flows are monitored from R_1. As the EI is increased to 10s the accuracy improves to the point where the 90th percentile of the error is under 0.1 for most hops.

6. DISCUSSION

Using inter-domain traffic: Our solution uses only intra-AS traffic for inference. However, in many networks, a significant proportion of the traffic is inter-domain traffic. The reason why inter-domain traffic is not suitable for inference is that it may contain markings from others ASes. Data packets may already be marked when entering an AS while ACK traffic can contain markings from many ASes on the forward path. Routers cannot separate the information related to the current AS just by observing the markings. Fortunately, separating inter-domain from intra-domain traffic can be conveniently done by ensuring that the source and destination addresses of a packet belong to the local AS.

Incremental deployment: Universal deployment of AQM and ECN is not necessary for our approach to be effective. Our solution can be incrementally deployed. It can be deployed on specific AQM/ECN enabled paths in the network. Moreover, it can be deployed around regular routers. The only condition is that the position of the regular routers be known by all other routers in the network. Their presence can be factored out by the algorithm because their marking rate is effectively zero. Our approach can also be used in heterogeneous environments. We do not require routers to use the same AQM algorithm nor the same parameters for the algorithm. All that we require is that congestion marking is used alongside an AQM algorithm.

Equal-cost multipath: For our solution to behave correctly, a router must precisely compute the paths taken by data packets. When Equal-Cost Multipath (ECMP) is used, this condition may not hold. Typically with ECMP, the decision to assign a flow to a link of equal cost is deterministic but local to a router. Therefore, ECMP decisions might not be discernible outside the router that made them. Consequently, if from routing information or any other source a router realizes that data packets are routed using ECMP, it should not use those packets or their corresponding ACKs for inference since it cannot reliably identify the path they

took. However, if ECMP decisions are based on a set of rules known by all the routers (e.g. hashing source addresses), the congestion on equal cost links can also be inferred.

Re-routing: When failures occur and routers route around them, link-state information cannot instantaneously propagate throughout the network. Therefore, some routers not yet aware of the changes may still attribute old paths to flows. Consequently, those routers will have counters that contain stale information. The correct solution is for the routers to reset all counters affected by the change once the new link-state information becomes available. Since link-state information is reliably disseminated this is guaranteed to happen.

7. RELATED WORK

The Re-ECN protocol [5] is a method for holding flows accountable for the congestion they create. It requires a non-standard use of header bits for TCP receivers to convey path level congestion information upstream. The TCP sources use an extra header bit to mark the data packets according to the information received. Policers placed on forward paths of flows can then use the source markings along with the ratio of ECN CE markings to obtain upstream and downstream path level congestion information. Policers can then detect misbehaving flows that create more congestion than permitted by current conditions. In contrast, our solution does not require any changes to either protocols or end hosts. Moreover, the use of routing information allows us to obtain fine-grained link level congestion information whereas Re-ECN routers obtain aggregate path level information.

Most methods developed for inferring the congestion severity are designed for TCP end-hosts and help them change their data sending rate according to current network conditions. TCP end-hosts are concerned only with the overall quality of the end-to-end paths. As a result, most of today's algorithms convey path level congestion information [23, 1]. Other approaches [2] report only the most congested link on a particular path. Our solution goes beyond these methods and uses the path level measurements to allow the routers to obtain fine grained link level congestion information.

Pathchar [9] is an end-user based tool that measures link level characteristics. Its high level technique is similar to our own in that it uses partial path level information and algebraic computations to obtain link level information. However, because it requires a significant number of probes, Pathchar is not scalable enough to convey dynamic link level information to routers running distributed protocols.

8. CONCLUSION

We presented a novel approach that routers can use locally to passively infer a network-wide congestion map. The congestion maps can be easily built today using information already existing from widely-used, standardized protocols. The maps are continuously updated using congestion markings from existing traffic and therefore provide dynamic congestion information. We showed that the inference accuracy is good using a wide-range of experiments with varying degrees of congestion and traffic properties.

Acknowledgments

This research was sponsored by the NSF under CAREER Award CNS-0448546, CNS-0721990, CNS-1018807, Microsoft Corp., IBM Corp., and by an Alfred P. Sloan Research Fellowship. Views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of NSF, Microsoft Corp., IBM Corp., the Alfred P. Sloan Foundation, or the U.S. government.

9. REFERENCES

[1] M. Adler, Jin-Yi Cai, J.K. Shapiro, and D. Towsley. Estimation of congestion price using probabilistic packet marking. In *INFOCOM*, April 2003.

[2] L. L. H. Andrew, S. V. Hanly, S. Chan, and T. Cui. Adaptive deterministic packet marking. *IEEE Communications Letters*, 10(11):790–792, 2006.

[3] G. Appenzeller, I. Keslassy, and N. McKeown. Sizing router buffers. In *SIGCOMM*, August 2004.

[4] S. Athuraliya, V. H. Li, S.H. Low, and Qinghe Yin. Rem: Active queue management. *IEEE Network*, May 2001.

[5] B. Birscoe, A. Jacquet, C.C Gilfedder, A. Salvatori, A. Soppera, and M. Koyabe. Policing congestion response in an internetwork using re-feedback. In *SIGCOMM*, 2005.

[6] R. Braden. *RFC 1122 - Requirements for Internet Hosts - Communication Layers*, 1989.

[7] Cisco. IOS Software Releases 11.1 Distributed WRED.

[8] C.V.Hollot, V. Misra, D. Towsley, and Wei-Bo Gong. On designing improved controllers for aqm routers supporting tcp flows. In *INFOCOM*, April 2001.

[9] A. B. Downey. Using pathchar to estimate internet link characteristics. In *SIGCOMM*, 1999.

[10] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, 1993.

[11] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot. Packet-level traffic measurements from the sprint ip backbone. *IEEE Network*, November 2003.

[12] J. He, M. Chiang, and J. Rexford. Towards robust multi-layer traffic engineering: Optimization of congestion control and routing. *IEEE J. on Selected Areas in Comm.*, 25, 2007.

[13] J. Hughes, T. Aura, and M. Bishop. Using conservation of flow as a security mechanism in network protocols. In *IEEE Symposium on Security and Privacy*, May 2000.

[14] Juniper. JUNOSe 9.1.x Quality of Service Configuration Guide: RED and WRED Overview.

[15] A. Kvalbein, A. F. Hansen, T. Cicic, S. Gjessing, and O. Lysne. Fast ip network recovery using multiple routing configurations. In *INFOCOM*, April 2006.

[16] K. Lakshminarayanan, M. Caesar, M. Rangan, T. Anderson, S. Shenker, and I. Stoica. Achieving convergence-free routing using failure-carrying packets. In *SIGCOMM*, 2007.

[17] L. Le, J. Aikat, K. Jeffay, and F. D. Smith. The effects of active queue management on web performance. In *SIGCOMM*, 2003.

[18] D. Maltz, G. Xie, J. Zhan, H. Zhang, G. Hjalmtysson, and A. Greenberg. Routing design in operational networks: A look from the inside. In *SIGCOMM*, August 2004.

[19] A. Mizrak, Y. Cheng, K. Marzullo, and S. Savage. Fatih: Detecting and Isolating Malicious Routers. In *DSN*, 2005.
[20] K. Ramakrishnan, S. Floyd, and D. Black. *RFC 3168 - The Addition of Explicit Congestion Notification to IP*, 2001.
[21] A. Shaikh, L. Kalampoukas, A. Varma, and R. Dube. Routing stability in congested networks: Experimentation and analysis. In *SIGCOMM*, August 2000.
[22] M. Shand and S. Bryant. IP fast reroute framework. Internet Draft draft-ietf-rtgwg-ipfrr-framework-13.txt, October 2009.
[23] R. Thommes and M. Coates. Deterministic packet marking for congestion price estimation. In *INFOCOM*, 2004.
[24] Bo Zhang. *Efficient Traffic Trajectory Error Detection*. PhD dissertation, Rice University, Houston, TX, 2010.