Reproducing RTT Estimation Results Using Fixed Share Experts in ns-3

Kevin Woodward
University of California, Santa Cruz
Santa Cruz, CA, USA
keawoodw@ucsc.edu

***Abstract*—We aim to benchmark the performance of the Fixed Share Experts algorithm as compared to Jacobson's algorithm when applied to round trip time estimation in a variety of networks. This Fixed Share Experts approach is a lightweight machine learning schema that has shown success in network efficacy over several scenarios with direct and indirect evaluation metrics. In the original work it has yielded over 40% improvement than Jacobson's algorithm in the QualNet simulator [1] [2]. In this paper we implement the Fixed Share Experts algorithm and the related network scenarios in the ns-3 simulator and compare differences in results. The purpose of this is twofold; we can compare the performance of the two algorithms as well as compare the differences in the comparisons between both simulators. Through observation of the resulting metrics we discover that there are significant differences in the results between simulators.**

***Index Terms*—Machine learning, AODV, Ad hoc networks, Ethernet networks, Computer networks**

I. Introduction

Round Trip Time (RTT) estimation is a crucial mechanic for network efficiency in direct and indirect ways when TCP is used. While it is a relatively small piece of a network definition, a change in RTT prediction can have drastic implications on the performance of a network, especially one that is unstable or non-deterministic. This is due to how TCP uses the estimation of the RTT for congestion control and flow changes for example.

Fixed Share Experts is a relatively simple machine online learning algorithm proposed in 1992 by Mark Herbster and Manfred Warmuth in [3]. The significance of online learning here is high, as this allows our algorithm to adapt over time by learning through trials rather than attempting to learn all patterns through pretraining on a large dataset. This algorithm can be described as a single layer learning network. The inputs to this network are a chosen number of experts at a fixed value, all contributing to a prediction with an influence of their respective weights. The sharing portion comes from a post weight update step to propagate a normalization of all weights to assist poorly performing experts.

In an effort to reproduce the work in the source paper [1], we apply this Fixed Share Experts algorithm to RTT estimation which will be described in more detail in Section III. In both simulators Jacobson's algorithm, a moving average calculation, is implemented for RTT estimation. The importance of the estimated RTT is how heavily re-transmission time out (RTO) estimation depends on it. RTO estimation is crucial in preventing issues such as extraneously waiting for an acknowledgement for a dropped packet or over-aggressively re-transmitting a packet that was not dropped, both of which can affect several aspects of network performance.

The design, implementation, and evaluation of the above will all follow the source paper and will be discussed in the corresponding sections. Results will be supplemented with graphical aids. Finally we will provide a conclusion highlighting the overall results and providing a reasoning for the difference between the results of the source paper and the results gathered here.

II. Related Work

The related works for this topic heavily follow the related works of the source paper. We implore you to read the source paper for a broader overview of related works.

RTT estimation is classically known as the exponentially weighted moving average (EWMA). This is simply represented as the following equation:

$$EstimatedRTT = (1-\alpha)\cdot EstimatedRTT + \alpha \cdot SampleRTT \tag{1}$$

Where α is typically 0.125. This is used in computing the RTT variance as well as RTO:

$$RTTVAR = (1-\beta)\cdot RTTVAR + \beta \cdot |SampleRTT - EstimatedRTT| \tag{2}$$

$$RTO = \max EstimatedRTT + K \cdot RTTVAR \tag{3}$$

Where β is typically 0.25 and K is typically 4.

In our work, we apply Fixed Share Experts to only Equation 1 while leaving Equation 2 and 3 alone.

This Fixed Share Experts algorithm can be seen as a state diagram as seen in Figure 1, and can also be more concretely defined by Algorithm 1.

In the source paper, $N = 100$, $\eta = 2.0$, and $\alpha = 0.08$. For all results shown in this paper, we use these values for consistency. These definitions come from the source paper and are recreated in the ns-3 simulator in the scenarios defined in the next section.

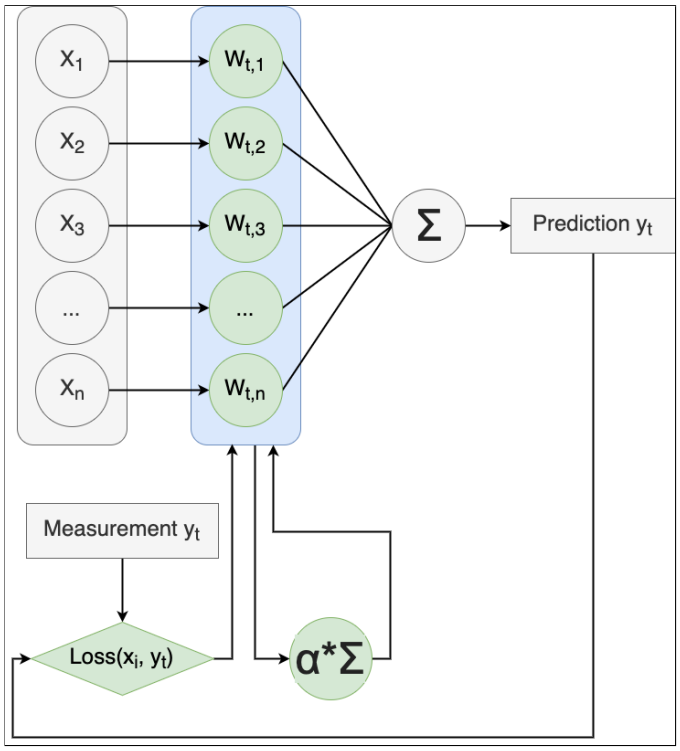

Fig. 1. State machine of Fixed Share Experts as applied to RTT estimation.

Algorithm 1: Fixed Share Experts as applied to RTT estimation algorithm

Parameters : $\eta > 0$ *and* $0 \leq \alpha \leq 1$

Initialization : $w_{1,1} = \ldots w_{1,N} = \frac{1}{N},\ x_i = RTT_{min} + RTT_{max} \cdot 2^{\frac{i-N}{4}}$

1) Prediction :

$\hat{y}_t = \frac{\sum_{i=1}^{N} w_{t,i} \cdot x_i}{\sum_{i=1}^{N} w_{t,i}}$

2) Computing Loss : $\forall i\ 1, \ldots, N$

$$L_{i,t}(x_i, y_t) = \begin{cases} (x_i - y_t)^2 & , x_i \geq y_t \\ 2 \cdot y_t & , x_i < y_t \end{cases}$$

3) Exponential Updates : $\forall i\ 1, \ldots, N$

$w'_{t,i} = w_{t,i} \cdot e^{-\eta \cdot L_{i,t}(y_t, x_i)}$

4) Sharing Weights : $\forall i\ 1, \ldots, N$

$pool = \sum_{i=1}^{N} \alpha \cdot w'_{t,i}$

$w_{t+1,i} = (1 - \alpha) \cdot w'_{t,i} + \frac{1}{N} \cdot pool$

III. Technical Approach

The approach consists of five network scenarios. These scenarios were designed by the source paper to observe how different parameters affect RTT estimation as well as other gathered metrics. Most of these scenarios share a significant portion of parameters, so these will be eschewed when possible for conciseness.

A. Scenario 1

This scenario consists of 20 wireless ad-hoc nodes in a two-dimensional simulation area of 1,500 by 1,000 meters over 25 minutes of simulated time. Each node follows a random waypoint mobility model to move within the confines of the simulation area. The initial position and waypoints of each node are randomized. The speed at which each node travels to its random waypoint is randomized between 1 and 50 meters per second with no pause time in between mobility waypoints. Each node is using the Ad-Hoc On Demand Distance Vector (AODV) routing protocol to determine valid routes. The transmission power of these nodes is limited such that the maximum transmission range is 100 meters. Both MAC and physical layers conform to the 802.11b standard. The traffic is generated over a number of TCP flows sending FTP data, which is chosen from the set of (3, 7, 17, 34, 68, 100, 130). The TCP send and receive buffers are set to 16,384 bytes each. For each of these flow counts, the experiment is reran to observe how RTT estimation as a function of traffic density affects performance. Each flow is constituted of a random sender and receiver, with the only limitation being that a node cannot send to itself. Each flow is sending packets of size 512 bytes. The number of packets for each flow is randomized to between 1,000 and 100,000, and the time at which each flow begins is randomized to any point between the beginning and end of the simulated time.

B. Scenario 2

This scenario follows the same parameters as scenario 1, but with 10 nodes instead of 20. This effectively reduces the network density.

C. Scenario 3

This scenario follows the same parameters as scenario 1, but nodes have no mobility model. This scenario evaluates a fixed wireless network.

D. Scenario 4

This scenario follows the same parameters as scenario 1, but with several modifications. Instead of a flow sending a random amount FTP data at a random time, each of the 20 nodes chooses a receiver and sends 1,000 packets every 200 seconds over a 90 minute simulation time. Therefore the concept of fully random flows is irrelevant in this scenario, and the varying metric is instead speed. Three speed categories are defined as (1-10m/s, 20-30m/s, 40-50m/s). If a node does not send its 1,000 packets over the 200 second time span, these packets do not roll over into the next 200 second time span.

This simulates a high burst traffic environment where 20 flows are created and started simultaneously every 200 seconds.

E. Scenario 5

This scenario only follows the parameters from scenario 1 regarding how much and at what time data is sent. This scenario is a fixed wired network consisting of two nodes creating a bottleneck link, with three nodes linked to each bottleneck node for a total of eight nodes and seven links. This tested over a set of (70, 100, 130) flows. This results in an extremely congested network, testing RTT estimation as a function of delay due to congestion.

IV. Implementation

All simulation work done was implemented in the ns-3 simulator [4]. A variety of Python3 scripts were written to parse and process output.

A. Fixed Share Experts RTT Estimator

Implementation of a new RTT Estimator involves modifying the core 'internet' module in ns-3. Since the existing RTT estimator class is already developed to be extended and its usage within the simulator is baked in, the obvious choice is to simply modify what exists. Conveniently, RttEstimator is in an existing base class from which the existing Jacobson's algorithm estimator derives from, which is called RttMeanDeviation in the source code.

Deriving from the RttEstimator base class to create the new RttFixedShare allows for minimal alteration to the source code wherever an RttEstimator type object is used. This is extremely convenient, as the implementation is as simple as deriving from the base class, implementing the declared virtual functions, and adding the required state necessary as member variables within our new derived class.

The core of the work is done within a single function called Measurement. This function is triggered whenever a socket receives an actual RTT measurement. Within the RttMeanDeviation implementation, this updates a member holding the RTT estimation and variance. Exactly the same is done in the RttFixedShare implementation of Measurement, but using the logic of the Fixed Share Experts algorithm instead, where the experts and weights are stored as member vectors.

Member values for the learning rate, sharing rate, and number of experts are also added in the RttFixedShare implementation, as well as an initialization function to perform expert spacing and weight initialization which is called within the constructor.

Enforcing the usage of the new estimator is as simple as including a configuration call within an ns-3 simulation script file. This configuration call sets which estimator object type the TcpL4Protocol should use by default.

The sum of implementing the new estimator only touches three files within the ns-3 core, which are rtt-estimator.cc, rtt-estimator.h, and rtt-test.cc. Modifying the test is necessary for ns-3 to compile after modification of the other two files.

B. Scenarios

The scenarios implemented were created in order. The only scenario not implemented is scenario 5 due to time constraints.

The example named manet-routing-example.cc included in ns-3 is extremely helpful in building scenario 1, and consequently the rest of them. It consists of wireless nodes using OnOffApplication objects to send UDP data with a configurable routing protocol, which includes AODV.

The actual simulation logic is not extremely complicated. Many helpers are used to instantiate and install the different layers onto nodes, as well as to generate and receive traffic. Configuration calls are simple and used to set fixed parameters across all scenarios. Determining sender to receiver flow pairs is done using the C++ rand function with paired with modulo to receive a uniform distribution.

For scenarios 2 and 3, only minor source code changes are needed after we implement scenario 1. For example, scenario 2 can be created from scenario 1 simply by changing the parameterized number of nodes in the source code and scenario 3 can be created from scenario 1 by simply removing the random waypoint mobility model and associated code.

Scenario 4 requires more modification than scenarios 2 and 3. Scenario 4 dictates that a new flow is scheduled every 200 seconds for each node. This involves creating a nested loop to generate the flow for each of the 20 nodes for every 200 second interval and schedule them accordingly.

C. Data Gathering

Gathering data for the evaluation metrics was done in several ways. These metrics are mean RTT error, goodput, delivery ratio, retransmission ratio, and congestion window size. The definition and relevance of these metrics will be discussed in further detail in the next section.

Collecting the error between an actual RTT measurement and the estimated measurement is done directly in the RttFixedShare implementation through use of the ns-3 log functionality. The stylistically correct way to gather this data would be to create a trace point to the desired variables and attach a callback in to the trace in the simulation code. However, nothing to create that trace would work, and ns-3 would throw a run-time exception when attempting to attach said callback.

Goodput, delivery ratio, and retransmission ratio were all obtained by the flow monitor utility that ns-3 provides. Flow monitor creates a markup (XML) file of every flow that runs in a given simulation. This includes explicitly created TCP data flows as well as AODV UDP routing flows. The contents of this flow monitor file include IPv4 and IPv6 classifiers and flow statistics. These flow classifiers can be used to filter out irrelevant flows and then the three aforementioned metrics can be gathered from the corresponding flow statistics. These statistics include valueble information such as total packets sent and received, first and last packets sent and received, and number of packets dropped.

Congestion window size was obtained by creating a trace per flow. This trace has a callback which writes the congestion window size in bytes whenever it changes. All flows write to

the same file to make parsing and averaging over all flows simpler.

D. Data Processing

Data gathering was done in a way to provide ease of parsing. The gathered data for each run was parsed with Python3 scripts to extract and apply a weighted average to the numeric values. These values were then recorded and charted in a spreadsheet to provide figures.

V. Evaluation Methodology

The direct comparison between Jacobson's algorithm and Fixed Share Experts is done over a series of averages. For every scenarios, 5 repetitions are ran and the results of these runs are averaged with a weight of the total packets transmitted over the sum of the flows. For each metric within a single run, said metric is a weighted average over each flow where that metric is gathered, again with packets being the weight. The only exception to this is the congestion window size, as the data gathered is already over all flows.

The definition of the collected evaluation metrics are as follows. Mean RTT error is the absolute difference in a measured RTT and the predicted RTT and is represented in milliseconds. Goodput is defined as packets received at the destination of a flow over the duration of the flow, with a unit of packets per second. Delivery ratio is simply a ratio of packets that were successfully received at the destination divided by the number of packets sent at the source, which is represented as a ratio. Retransmission ratio is the ratio of packets that were retransmitted divided by the number of packets sent at the source, also represented by a ratio. Finally, congestion window size is the size of the congestion window over all RTT prediction trials, and is represented in bytes.

The evaluation metrics fall into two categories, direct and indirect results. The only direct metric is the mean RTT error. Out of all collected metrics, only this metric directly measures the performance of the chosen RTT estimator. While just as significant, all other metrics represent an aspect of the performance of the network as a function of the chosen RTT estimator, therefore we describe them as indirect.

The main purpose of evaluation is to compare the comparisons between the source paper and the ns-3 results. For brevity, we only include figures from the source paper regarding the mean RTT error as this is the most comparable result. The other metrics are less comparable due to the differences in the QualNet and ns-3 internals.

VI. Results

As mentioned, we provide the mean RTT error figure from the source paper for reference for each scenario. As a blanket statement, every metric in the source paper has a very decisive trend clearly showing that the Fixed Share Experts approach performs better than Jacobson's algorithm. This not only holds true for the included figures from the original paper, but also for metrics that we have not included graphs for.

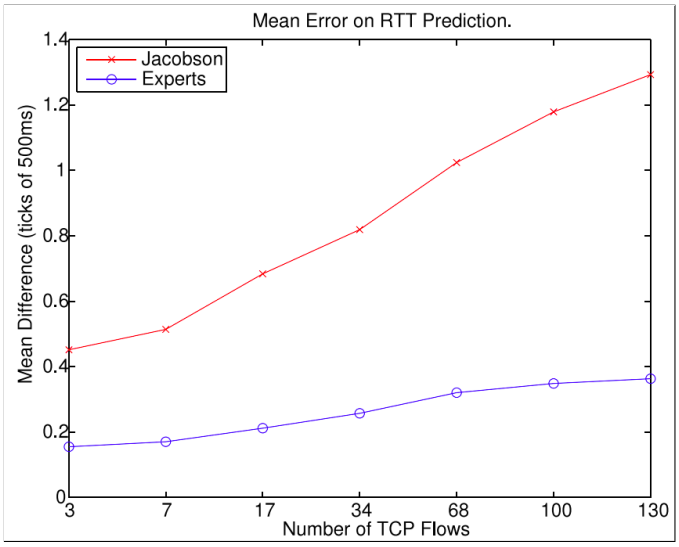

Fig. 2. Mean RTT error between predicted and measured for scenario 1 in QualNet.

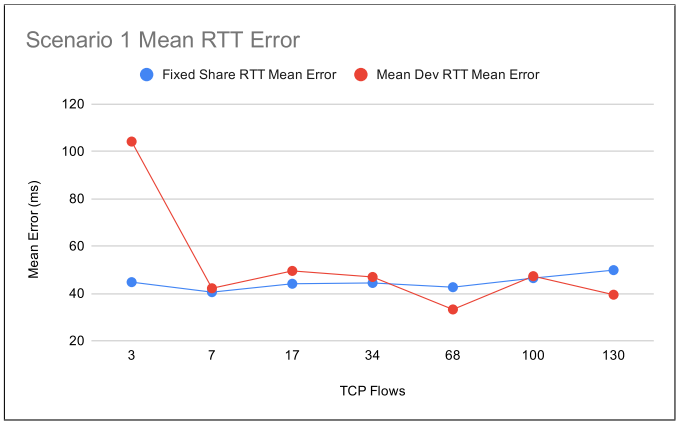

Fig. 3. Mean RTT error between predicted and measured for scenario 1 in ns-3.

Comparing values directly between the source paper and the ns-3 results is not accurate, as the measured RTT values are different as well as other simulator internals. The more accurate comparison is the difference between both algorithms results between QualNet and ns-3.

A. Scenario 1

This scenario deals with 20 moving moving nodes. As is visible in Figure 2, we see an upward trend in error in both algorithms in QualNet with the trend being more severe with Jacobson's algorithm. Fixed Share Experts still trends upwards but can react to changes in measured RTT values more quickly, thus flattening the curve significantly.

The trends resulting from the ns-3 data in Figure 3 are much more sporadic than from QualNet. This is true for the result of the results as well. Fixed Share Experts follows a similar trend to the QualNet results with some fluctuation in the lower flow counts. Interestingly, Jacobson's algorithm produces a massive spike at three flows. We believe the sparsity of the flows is

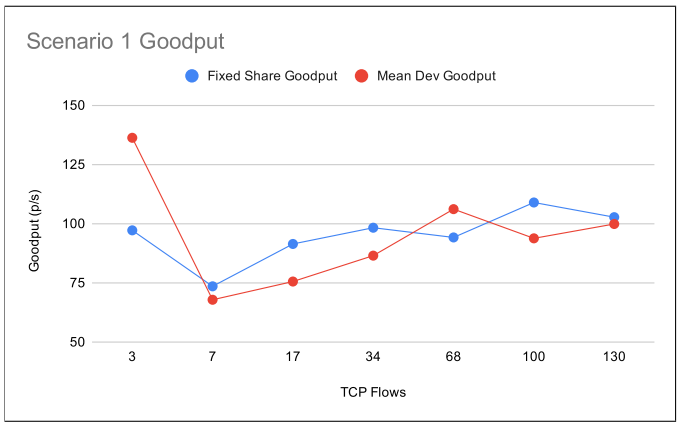

Fig. 4. Received packets at the destination over time for scenario 1 in ns-3.

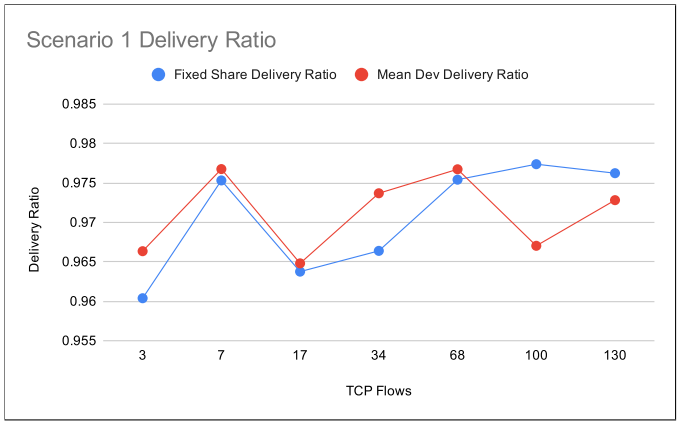

Fig. 5. Ratio of delivered packets over all packets for scenario 1 in ns-3.

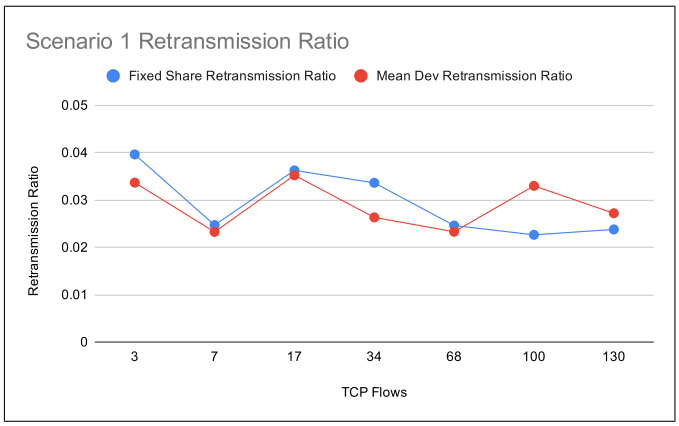

Fig. 6. Retransmitted packets over all packets for scenario 1 in ns-3.

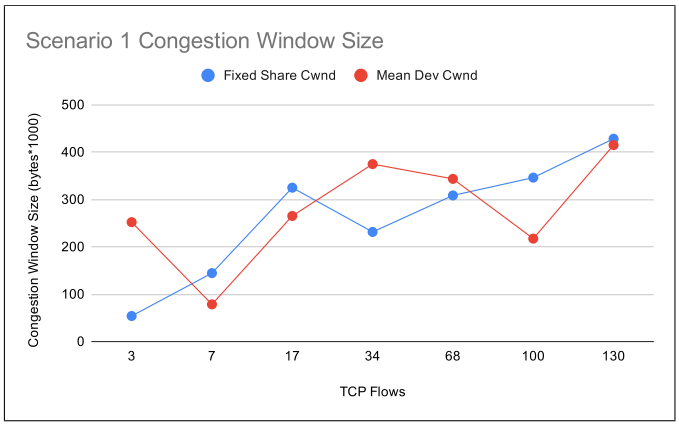

Fig. 7. Average congestion window size for scenario 1 in ns-3.

causing a high average as the large amount of randomness in the experiment can lead to often invalidation of routes.

For goodput in Figure 4 we expect a steep increase as the number of flows increase when observing the source paper. The spike at three flows is likely due to the minimal amount of route collision, as any node handling more than one flow will experience a loss in goodput as the FTP data saturates the transmission. Through all simulations, a peak of 137 packets per second was achieved for any single flow. The source paper shows an average near 300 packets per second for higher flows. The reason for this discrepancy is due to the transmission rate being set to 1Mpbs in ns-3 for a wider range of RTT values, as without this we see RTT measurements routinely less than 20ms. Lowering our transmission rate is an easy way to achieve more comparable results.

The results for the delivery ratio metric in Figure 5 follow essentially no trend when compared to QualNet. We see mostly noise in our ns-3 results, but with a few key insights. We can observe that there is a delivery ratio spike for both algorithms at 7 flows. We believe this number of flows provides a good balance of enough flows for sufficient connectivity and also few enough flows to not over congest nodes. At higher flows, there is not a clear decision for which algorithm performs better for this metric.

For retransmission rate in Figure 6, we again see this spike in performance at 7 flows. Interestingly, the best performing retransmission ratio at 68 and 100 flows for Jacobson's algorithm and Fixed Share Experts respectively is within less than half of a percent of the value at 7 flows. This implies the noise on this metric is high when compared to the actual trend, and further implies that the RTT estimation method has a low impact on retransmission rate for this scenario.

For QualNet, we see an inverse relationship between flow count and congestion window size. However this relationship is the opposite for our ns-3 results in Figure 7. This is an interesting difference between simulators. We believe less flows than desired are actually being established in ns-3. As a consequence of this, the increased flows simply are allowing the average congestion window size to become more significant in samples and thus actually grow linearly when past the slow start threshold.

B. *Scenario 2*

This scenario is identical to scenario 1, but with only 10 nodes. In Figure 8, we can see the QualNet mean RTT error trend is very similar to scenario 1, but more aggressive in the higher flows.

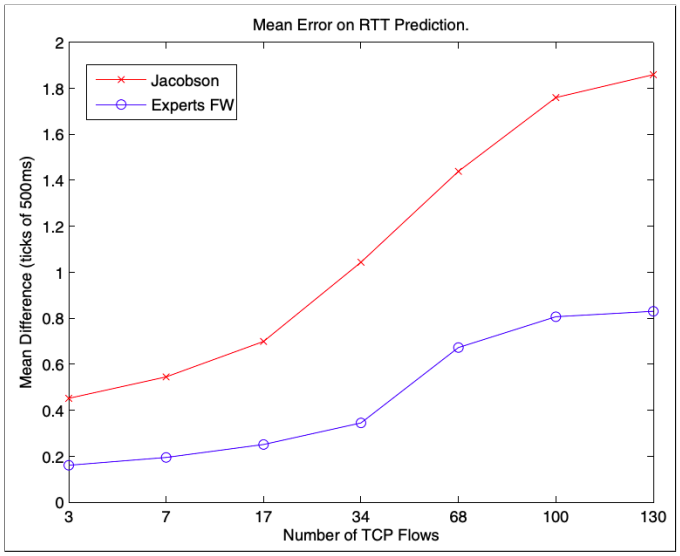

Fig. 8. Mean RTT error between predicted and measured for scenario 2 in QualNet.

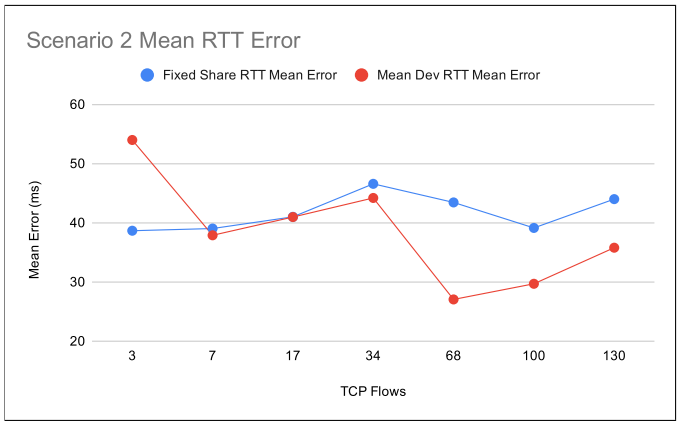

Fig. 9. Mean RTT error between predicted and measured for scenario 2 in ns-3.

For comparison to ns-3, we observe Figure 9. Our Fixed Share Experts trend is relatively flat, but our Jacobson's algorithm trend generally decreases as flows increase, with the biggest drop being between 34 and 68 flows. The decreased density of the network clearly affects this and has a clear explanation. The number of flows per node will be doubled over scenario 1 for any given flow count.

Goodput for scenario 2 also exhibits a downward trend, again opposite to QualNet seen in Figure 10. This trend does not exactly follow the mean RTT error trend, implying there is another factor at play. Also note that at lower flows, Fixed Share Experts performs better than Jacobson's algorithm contrary to scenario 1.

Delivery ratio has little information to glean in Figure 11. Fixed Share Experts exhibits a slightly downward trend, whereas Jacobson's simply is fluctuating around roughly 0.975.

Since retransmission ratio and delivery ratio are very closely

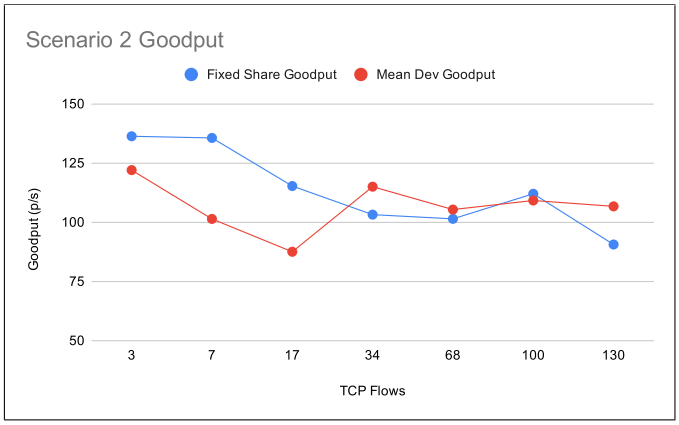

Fig. 10. Received packets at the destination over time for scenario 2 in ns-3.

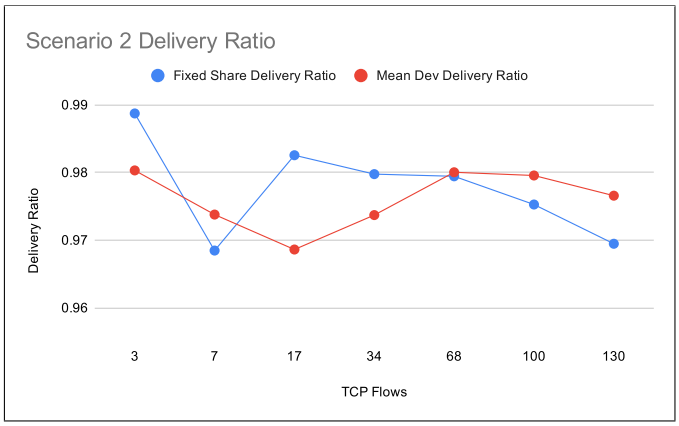

Fig. 11. Ratio of delivered packets over all packets for scenario 2 in ns-3.

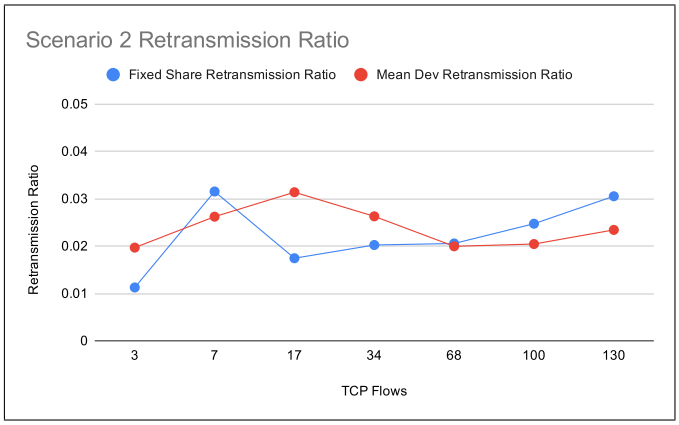

Fig. 12. Retransmitted packets over all packets for scenario 2 in ns-3.

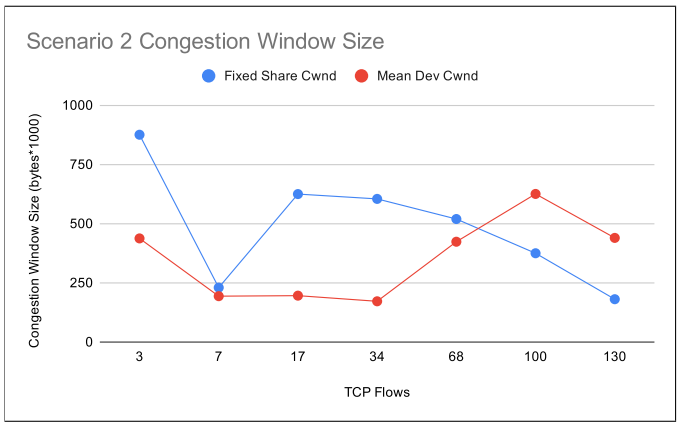

Fig. 13. Average congestion window size for scenario 2 in ns-3.

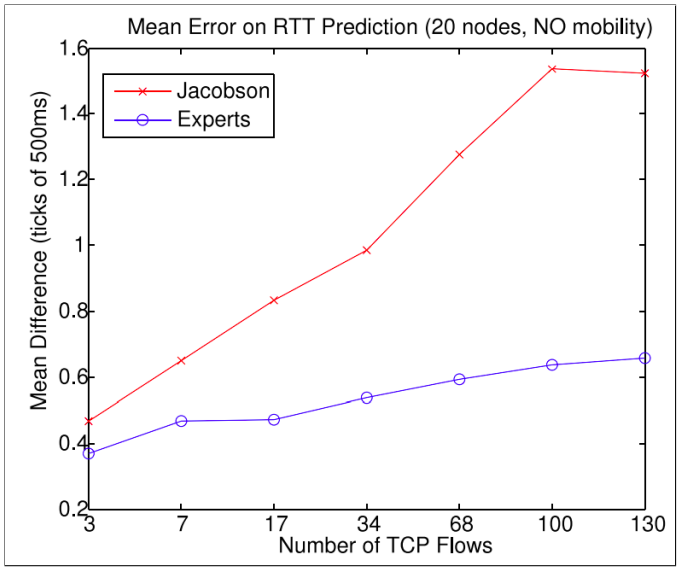

Fig. 14. Mean RTT error between predicted and measured for scenario 3 in QualNet.

related, Figure 12 is also of little help for insight.

The congestion window size in Figure 13 provides more information. We see that Fixed Share Experts has a relatively strong trend downward aside from the dip at 7 flows. We believe this can also be accounted for by the relative increase in node congestion. The congestion window size for Jacobson's algorithm is not convincing, but it increases in the higher flows.

C. Scenario 3

This scenario deals with a 20 node stationary network. The disparity between the two trends in Figure 14 is much larger than in the former two QualNet simulations. Jacobson's algorithm flattens out at the top of the graph, showing that peak congestion occurs before 130 flows.

In Figure 15, we see a very decisive set of trends for mean RTT error. However, it is the opposite of what we would expect. The error for Jacobson's algorithm is consistently

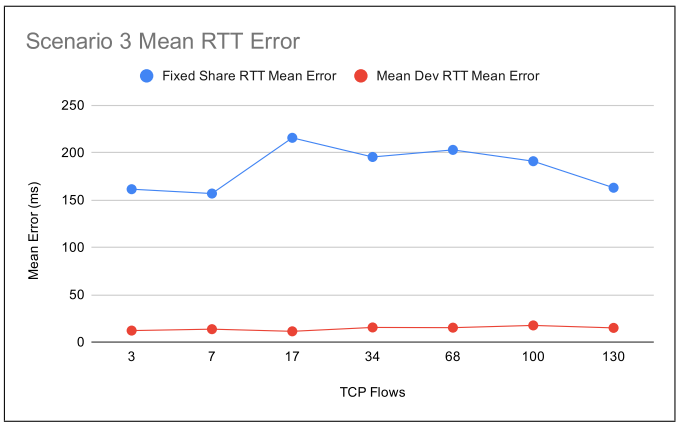

Fig. 15. Mean RTT error between predicted and measured for scenario 3 in ns-3.

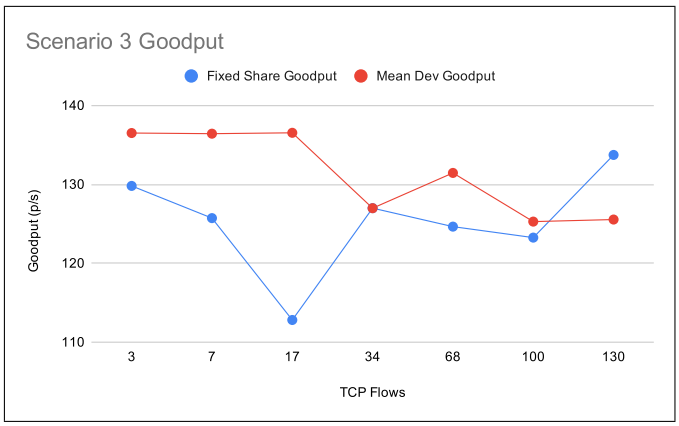

Fig. 16. Received packets at the destination over time for scenario 3 in ns-3.

extremely low, unaffected by the number of flows. The error for Fixed Share Experts is relatively consistent, but extremely high. The relative error differs by roughly a factor of 10. Such a remarkable difference is bizarre, especially considering the other metrics in scenario 3 are much more comparable. Unlike in the source paper, it does not seem like we are saturating our send and receive buffers, which would lead to an increase in mean RTT error as flows increase. This completely depends on what flows are established due to the the random initialization of position of the nodes.

In contrast to the mean RTT error for this scenario, the goodput in Figure 16 is within the same order of magnitude. Jacobson's algorithm beats Fixed Share Experts in most cases, which is not similar the QualNet results but is understandable. The lack of mobility removes a large portion of the RTT variability. This explains why the trends for mean RTT error are very flat, and why Jacobson's algorithm outperforms Fixed Share Experts in this scenario.

For delivery ratio in Figure 17, we are seeing extremely high values when compared to scenarios 1 and 2. Again, this is attributed to the stationary nodes where routes will not be broken, and routes that can never connect will not establish flows, thus not contributing to the statistic. We have also

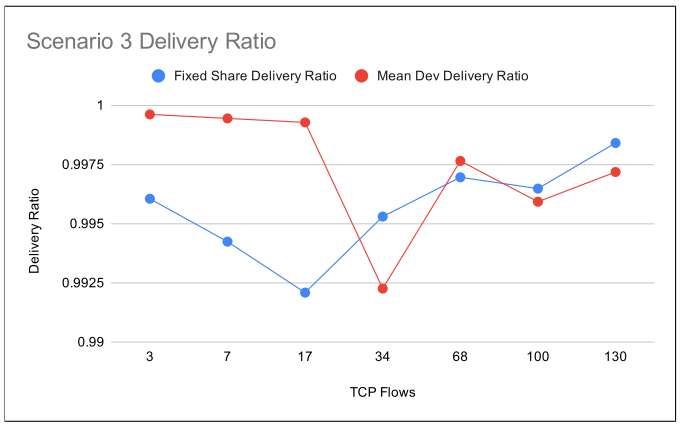

Fig. 17. Ratio of delivered packets over all packets for scenario 3 in ns-3.

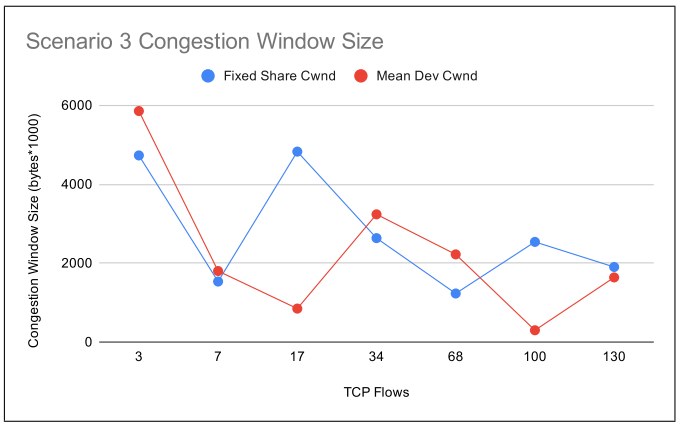

Fig. 19. Average congestion window size for scenario 3 in ns-3.

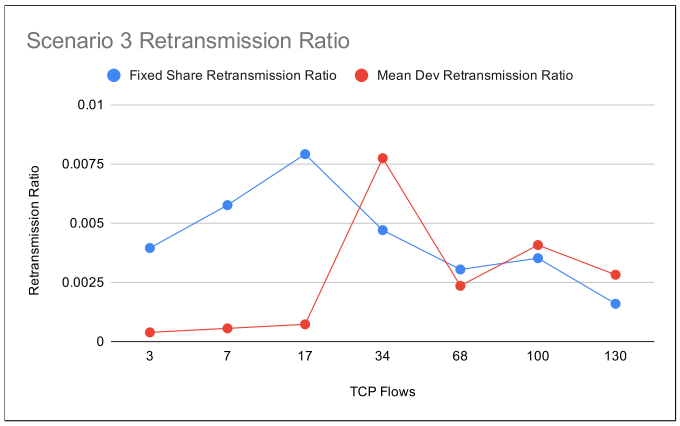

Fig. 18. Retransmitted packets over all packets for scenario 3 in ns-3.

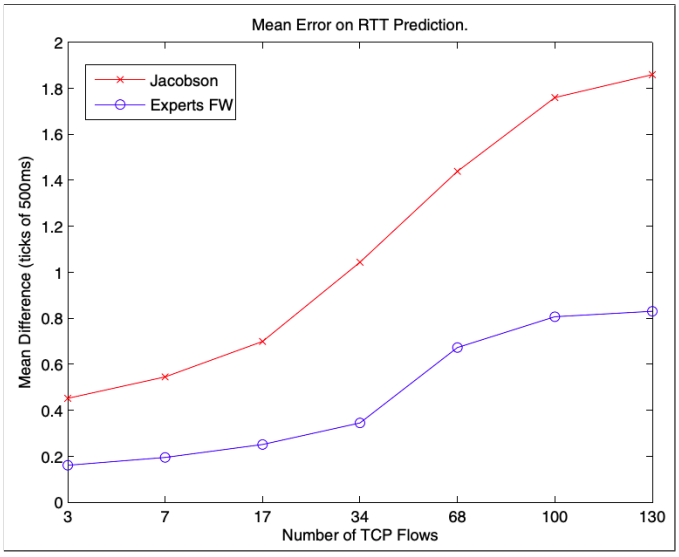

Fig. 20. Mean RTT error between predicted and measured for scenario 2 in QualNet.

observed that we are not saturating our buffers, which would cause a lower delivery ratio as packets would be dropped from the queues.

We can also see a similar story with our retransmission ratios in Figure 18. While it may appear to fluctuate a significant amount, all values are well under 1%. The fluctuations at this consistently low of a value can be attributed mostly to noise due to random simulation conditions. In fact, in the source paper this metric is eschewed for this scenario in favor of an average queue length metric, which was not collected. Both delivery and retransmission ratio results are better in the ns-3 implementation than the QualNet implementation.

The congestion window trends seen in Figure 19 vaguely follow the trend of the source paper's respective figure. As we have seen commonly, there are certain flows where either algorithm performs better. Note the range of values this figure occupies, which are much lower than the other scenarios for both simulators.

D. Scenario 4

Scenario 4 is evaluated over a range of speeds rather than a number of flows. This is observed in Figure 20. Interestingly, the best mean RTT error in QualNet occurs at the average speed of 25 meters per second for both algorithms, although the Fixed Share Experts estimator presents much less variation as the source paper mentions.

Unlike the QualNet results, we see an inflection point in Figure 21 of our ns-3 results. Instead of both estimators seeing a minimum at 25 meters per second, Jacobson's algorithm has an increase in error as time increases and Fixed Share Experts is the opposite. Also note that the amount of variability between both algorithms is close to identical. We expect Jacobson's algorithm to progressively get worse over increasing speeds, but we do not expect Fixed Share Experts to be worse at slower speeds. It is uncertain as to what causes this and further research is needed.

The rest of the results for this scenario are extremely similar between both simulators. Observe in Figure 22 that Fixed Share Experts is only very marginally better than Jacobson's algorithm in terms of goodput. There seems to be a difference in how data is collected for goodput. The values in the source paper are several orders of magnitude larger, which seems

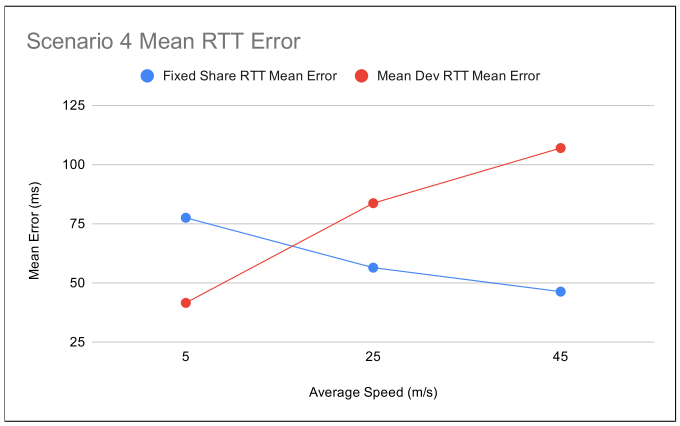

Fig. 21. Mean RTT error between predicted and measured for scenario 4 in ns-3.

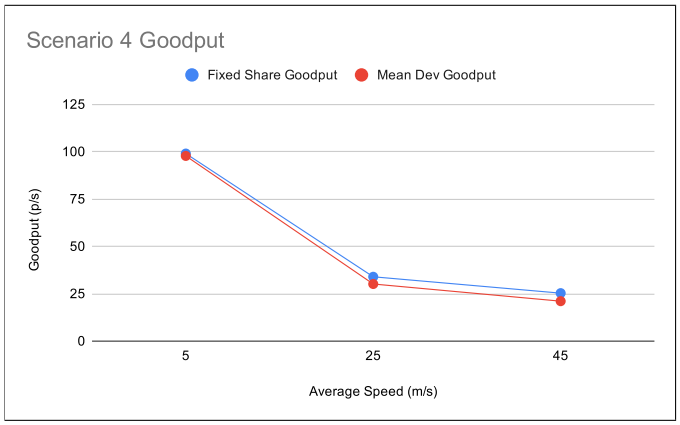

Fig. 22. Received packets at the destination over time for scenario 4 in ns-3.

to imply goodput is computed differently somehow, as this occurrence is only for scenario 4. The trend is similar with the exception of the values at the 45 meters per second mark.

Delivery ratio is all around lower for both estimators in ns-3 as seen in Figure 23. This is unlike the QualNet results, where an average of roughly 0.96. This could potentially be caused

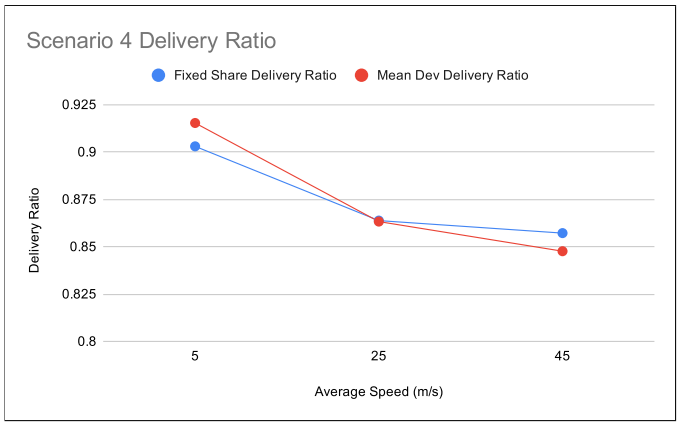

Fig. 23. Ratio of delivered packets over all packets for scenario 4 in ns-3.

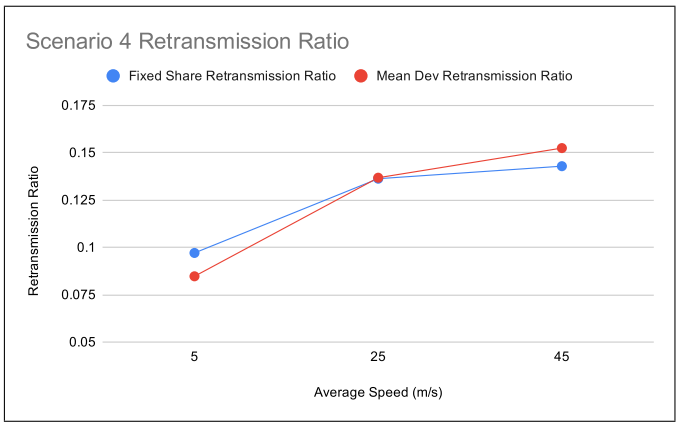

Fig. 24. Retransmitted packets over all packets for scenario 4 in ns-3.

by a difference in link speed. As mentioned, the link speed was set to 1Mbps for more statistically significant results. However, the source paper does not specify the link speed, which would have an impact in a bursty traffic scenario. A higher link speed would result in less send and receive queue saturation, therefore increasing delivery ratio.

Again, the values for retransmission ratios in Figure 24 are lower in QualNet than for ns-3. We attribute the same aforementioned reasons to this.

VII. Lessons Learned

As is visible from most of the results, there are large discrepancies in these simulations done in QualNet versus ns-3. There are several parameters that are left unspecified in the source paper, most of which are implementation details that differ between the two simulators. For example, AODV has parameters for number of route request (RREQ) and a valid route timer present in both simulators. Since this was not specified in the source paper the values were left as default in ns-3, as the difference in simulators is just as important as the comparisons. After the experiments were ran, it was found that these parameters differ significantly between the simulators. In QualNet, the maximum amount of RREQ retries allowed before being added to a blacklist is 3, whereas in ns-3 it is 2. The timer for how long a route is considered valid is 400ms in QualNet, whereas it is 3s in ns-3. Differences such as these can significantly affect how often, for how long, and which nodes connect in a mobile ad-hoc wireless situation when using AODV.

Differences such as these are what differentiate the simulators, and more comparable results to QualNet could be achieved by modifying every aspect of ns-3. While this may have been more validating, it does not provide an accurate representation for the goal of this paper.

VIII. Future Work

The most pressing work to be done is to finish creating and gathering data for scenario 5. While we know how well our estimator performs in mobile wireless ad-hoc scenarios, we do not have any data for a wired network.

Following this, we would like to rerun all scenarios with a higher number of averaging runs. In the source paper, each scenario flow combination was ran 24 times, whereas we only performed 5 due to time constraints. We believe this is contributing a significant amount of noise in our data, especially on random heavy scenarios such as 1 and 2. Information can also be further extracted from the collected data by computing the standard deviation for each result.

In observing scenarios with high measured RTT variance such as scenarios 1 and 2, we notice that the measured RTT values routinely spike to extremely high values on the order of several seconds for a short period of time. Since the prediction portion of Fixed Share Experts is simply a weighted average of the expert values, it is impossible for a prediction to be higher than RTT_{max}. When a measured RTT value of higher than RTT_{max} is observed, losses for every expert will be computed to be identical and very high, as observed by Algorithm 1. This destroys any accuracy the estimator has, and regresses the measurement to the unweighted average of the expert values. Even worse, the weights will all regress to very small values, causing recovery time to be high for the estimator to contribute accurate estimates in the future. To combat this, we propose a modification to the Fixed Share Experts algorithm as applied to RTT estimation.

Algorithm 2: Proposed RTT_{max} update algorithm

Parameters : $\tau > 0$

if $RTT_{measured} > RTT_{max}$ **then**

$RTT_{max} = RTT_{measured}$

$x_i = RTT_{min} + RTT_{max} \cdot 2^{\frac{i-N}{4}}$

$w_{t,i} = w_{t,\lfloor i \cdot \frac{RTT_{measured}}{RTT_{max}} \rceil}$

else

$RTT_{max} = RTT_{max}^{-\frac{1}{\tau}}$

end

This algorithm allows Fixed Share Experts to adapt to measured RTT values higher than RTT_{max}. When this occurs, the expert spacing is recomputed and the weights are shifted to match the re-spacing of the experts. If the measured RTT is lower than RTT_{max}, we apply an exponential decay RTT_{max} with an exponent τ defining the aggression of the decay. Tuning of this new hyper parameter allows accurate tracking of RTT_{max} in a network with a high amount of measured RTT variation. In short, we do not treat RTT_{max} as a global constant, but as a shifting variable with an optimal value for any given time slice of a network.

IX. Conclusion

In this work we recreate and analyze simulation results applying Fixed Share Experts RTT estimation to multiple scenarios in ns-3 compared to equivalent experiments in QualNet. While we do not achieve as decisive results as QualNet, we gain important insights as to what the implemented RTT estimator affects, to what magnitude metrics are correlated, and which scenarios benefit the most from the implemented estimator. While Fixed Share Experts provides a performance increase over Jacobson's algorithm for all metrics in QualNet, we observe that there are cases in ns-3 for which either algorithm performs better, both in direct and indirect metrics.

For an exact comparison, we can confidently say that there are differences in simulator internals preventing Fixed Share Experts from providing results similar to QualNet. However, adjustment of hyper parameters such as η, α, and the proposed τ could provide better results in ns-3 than we are currently observing.

As mentioned in the source paper, determining optimal values for these hyper parameters is not trivial, and the optimal solution varies based on the network as well as the desired amount of computation. Moreover, determining what differences affect the gathered metric requires a deep understanding of the internals of both QualNet and ns-3. We believe this work provides an insightful comparison into the efficacy of Fixed Share Experts as applied to RTT estimation and to what aspects need to be further researched to more confidently claim the performance of Fixed Share Experts.

References

[1] B. A. A. Nunes, K. Veenstra, W. Ballenthin, S. Lukin, and K. Obraczka, "A machine learning framework for tcp round-trip time estimation," *EURASIP Journal on Wireless Communications and Networking*, vol. 2014, no. 1, p. 47, 2014.

[2] "Qualnet - network simulation." [Online]. Available: https://www.scalable-networks.com/qualnet-network-simulation

[3] M. Herbster and M. K. Warmuth, "Tracking the best expert," *Machine learning*, vol. 32, no. 2, pp. 151–178, 1998.

[4] ns 3. [Online]. Available: https://www.nsnam.org/