TRIO - A SCHEME FOR ACTIVE RESOURCE MANAGEMENT IN DIFFERENTIATED SERVICES NETWORKS

A Thesis

by

SAIKRISHNAN GOPALAKRISHNAN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2000

Major Subject: Computer Science

TRIO - A SCHEME FOR ACTIVE RESOURCE MANAGEMENT IN DIFFERENTIATED SERVICES NETWORKS

A Thesis

by

SAIKRISHNAN GOPALAKRISHNAN

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Riccardo Bettati
(Chair of Committee)

A.L. Narasimha Reddy
(Member)

Nitin Vaidya
(Member)

Wei Zhao
(Head of Department)

May 2000

Major Subject: Computer Science

ABSTRACT

TRIO - A Scheme for Active Resource Management
in Differentiated Services Networks. (May 2000)
Saikrishnan Gopalakrishnan, B.E., Bharathiyar University, India
Chair of Advisory Committee: Dr. Riccardo Bettati

Differentiated services architecture is receiving wide attention as a scalable mechanism for providing different levels of Quality-of-Service (QOS). Recent studies, both analytical and simulation-based, have shown that the provided bandwidth in such an architecture may differ from the contracted rates. In this paper, we propose an active resource management approach that brings the provided service closer to the target rates. The proposed approach, SACRIO, actively allocates available excess bandwidth through localized packet remarking within a router. It is shown that SACRIO is simple to implement within the diff-serv architecture. It is also shown that the packet handling cost remains O(1) with SACRIO. SACRIO is shown to be quite effective through simulations.

To Amma and Appa

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Riccardo Bettati for his continued guidance and support during the preparation of this thesis. Dr. Bettati's recommendations during the initial stages of the research were really helpful.

I am deeply indebted to Dr. A.L.N. Reddy for providing me with invaluable advice and guidance throughout this research. He created a congenial atmosphere through regular and informal meetings. I appreciate his help and patience when I faced problems. He gave me excellent opportunities to enhance my knowledge in the field of differentiated services network. It has been a pleasure working under him.

I thank Dr. Vaidya for his interest and suggestions in this research. I also appreciate the help extended at the Institute for Scientific Computation, TAMU for letting me use their resources for my thesis work.

Words cannot truly express my deepest gratitude and appreciation to my parents for their financial and emotional support to realize my dream of pursuing masters degree in computer science. Completion of this work would have been impossible without them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

With the advent of multi-media applications , the heterogeneity of the traffic on the internet has increased. What used to be a medium for passing text oriented data is now the infrastructure for exchanging data, voice and video. Applications based on audio and video may require delay and throughput guarantees. In order to support these new applications, much work has been done in devising mechanisms for providing network QoS.

Considerable amount of work has been done in scheduling for the realization of throughput and delay guarantees for the flows [1, 2, 3, 4]. These mechanisms need to maintain per-flow state information of some sort. As, the routers need to switch millions of flows per second, maintaining per-flow state may not be scalable. Also, [5] suggests that scheduling without any support from buffer management results in ineffective sharing of bandwidth.

Diff-serv architecture is proposed to provide different levels of services [6, 7]. Diff-serv architecture does not employ per-flow state and hence expected to have better scaling properties. In a diff-serv network, routers are divided into two categories viz., Edge routers and the Core Routers as shown in figure 1. The core routers maintain no state information. This makes the network scalable since the routers that maintain full state viz., the edge routers, unlike the core routers, have only a limited number of flows going through them. The Edge routers have the components called the *markers* and the core routers have *droppers*. Essentially the markers mark the headers of the packets depending upon certain factors that are discussed below and the core routers

---

The journal model is *IEEE Transactions on Automatic Control*.

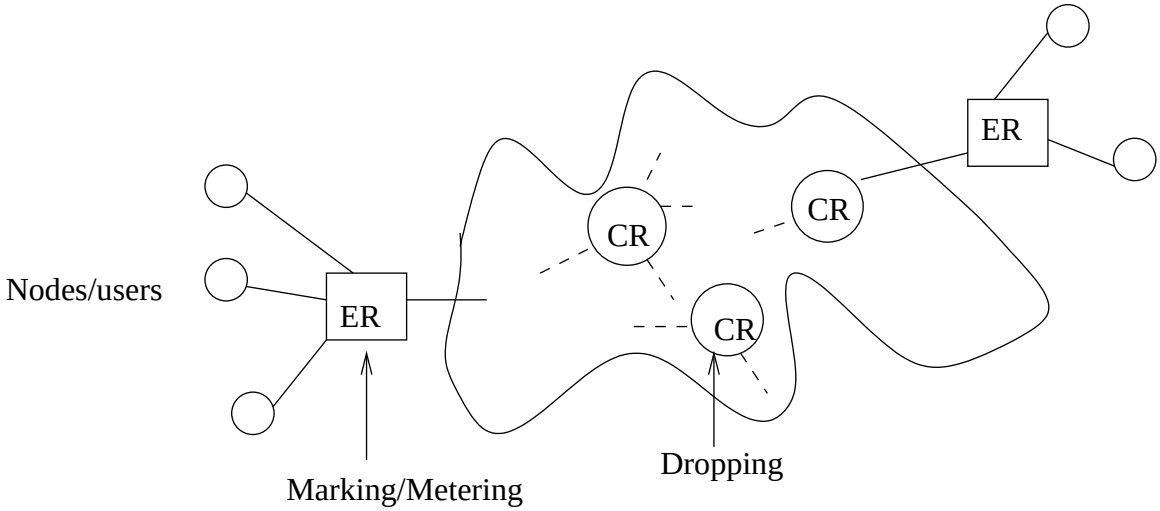

Fig. 1. Differentiated services network

recognize the marking that are done by the edge routers and use that information for resource management.

Considerable amount of work has been done in realizing the target rates in a diff-serv network by deploying different marking and dropping strategies . One such marking and dropping strategy is discussed below. Based on the contract (assured throughput guarantees for example), the packets are marked IN if the flow is in-profile and OUT if the sending rate of the flow is out of profile. At the time fo congestion, the core router drops the OUT packets first. Figure 2 explains this graphically. This two-drop precedence is first proposed in RIO (RED with IN/OUT) [7]. Discussion on RED (Random Early Detection) can be seen in [8]. Like the two-drop precedence discussed above, three-drop precedences also have been studied in [9, 10].

Consider a core router with a capacity C at its output link L. Let N be the number of instantaneous flows going through that router on to its output link L. Let R_1,R_2,R_3,...R_n be the corresponding reserved rates of flows F_1,F_2,...F_n. The excess

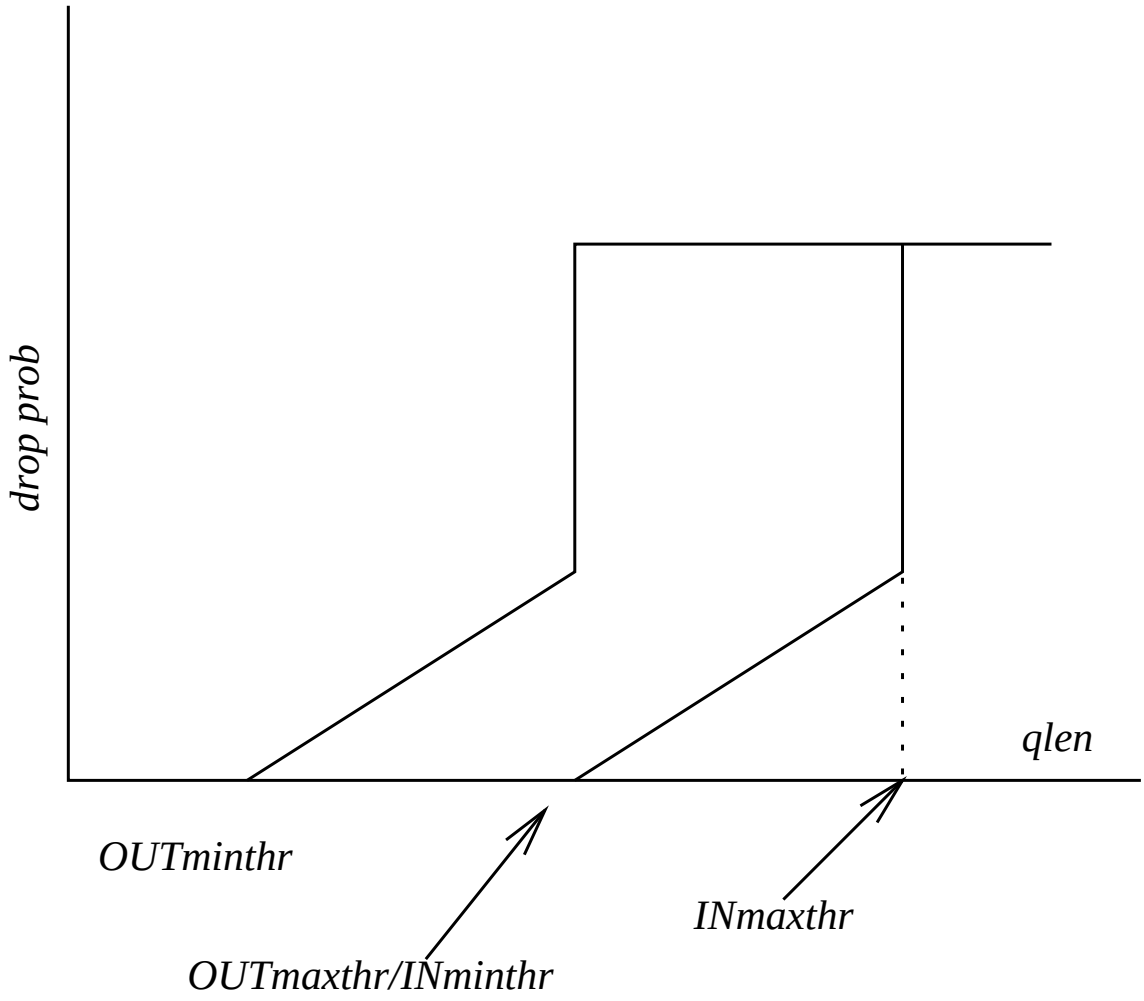

Fig. 2. RIO with IN/OUT

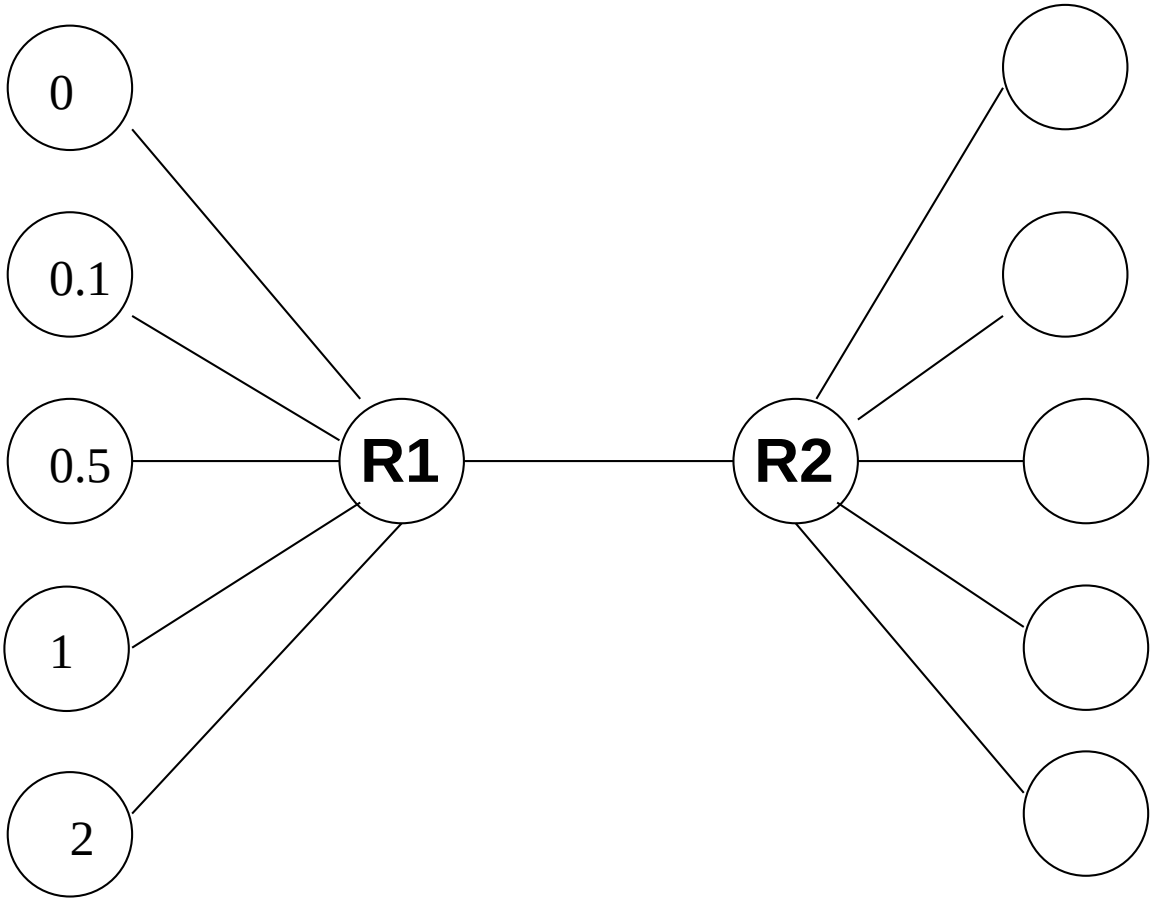

Fig. 3. Network topology for simulations

bandwidth E that is left over the link L is given by,

$$E = C - \sum_{i=1}^{n} R_i \tag{1.1}$$

Let $\alpha = E/N$. If the excess bandwidth is shared equally, the target rate T_i of every flow will be

$$T_i = R_i + \alpha \tag{1.2}$$

In [11] the author has analyzed plain RIO from the perspective of realizing throughput guarantees in a differentiated services network. Figure 3 shows a simple network topology that is used for the simulations. Each circle in the figure stands for two nodes with reservation mentioned in Mbps inside the circle. The marker uses

a sliding window leaky bucket marking strategy proposed in [7]. The router uses RED parameters 20/40/0.5 for the OUT packets and 50/100/0.02 for the IN packets. The total allocated bandwidth is 7.2 Mbps. The link bandwidth is set at 12 Mbs and 8Mbs in two different experiments to simulate allocations of 60%, 90%, of capacity. All the flows are assumed to have the same RTTs of 40 ms and run for 30 seconds.

While the ideal target rates being the ones that are calculated by using equation 1.2, the results of the simulation are shown in figures 4 and 5

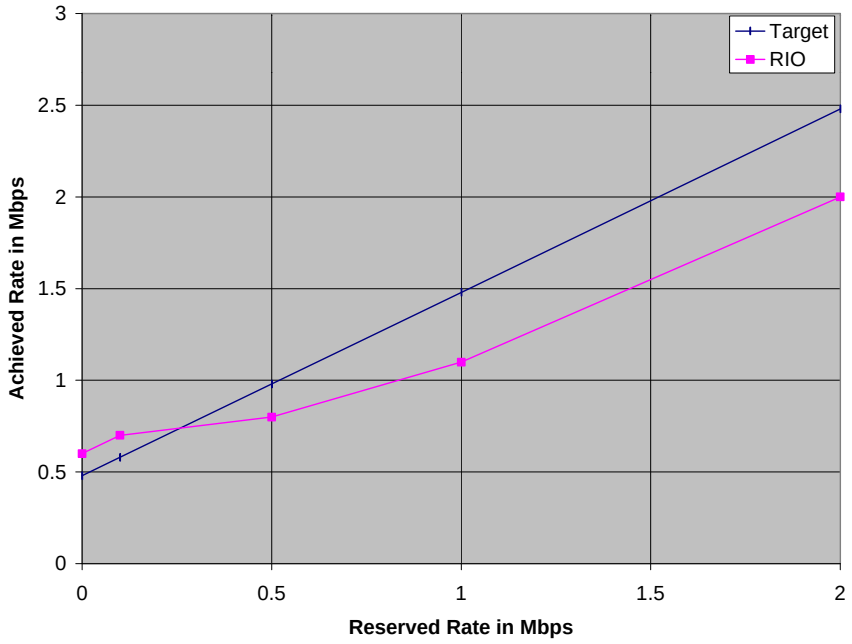

Fig. 4. Realized rates at 60% subscription

In the figures 4 and 5, the achieved throughput is plotted against the ideal target rates. The realized throughput is conspicuosly different from the ideal target in most

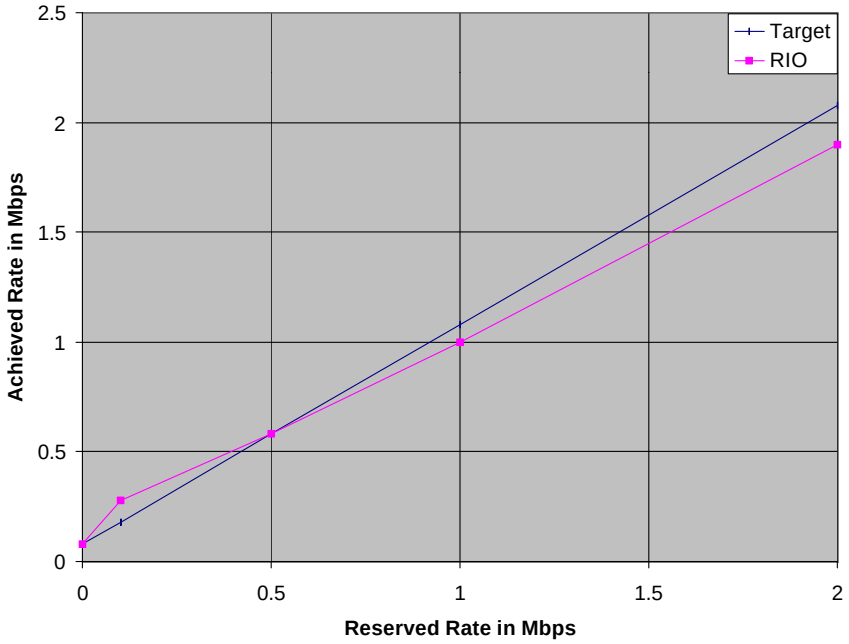

Fig. 5. Realized rates at 90% subscription

of the cases.

Recent simulation studies [11, 12, 13] have made the following observations about the realized rates. The flows with smaller targets exceed their target rates whereas the flows with larger targets do not achieve their target rates.

Since TCP has a multiplicative decrease and additive increase response to congestion, it results in different recovery times to regain the original congestion window for different flows . The flows with larger reservation have longer recovery times to regain their original rate. Hence the flows with smaller target rates exceed their targets and flows with larger targets may not achieve their targets. Also a flow with a

smaller RTT recovers faster than a flow with a larger RTT.

Simulation studies have also shown that non-responding flows can claim significantly larger bandwidth than their reserved rate [11]. Such non-responsive flows punish congestion-aware responsive TCP flows. Many researchers have pointed to the need of containing such non-responsive flows [14]. This problem, though less severe in diff-serv architecture, remains to be addressed.

Simulation results have also pointed to the difficulty of equalizing the realized rates even when all the flows are responsive TCP flows. It has been shown that flows with smaller RTT can realize larger bandwidths than flows with larger RTT.

Recent analytical work has also shown that it is not possible to realize target rates with RIO [15, 16]. In this thesis an active resource management mechanism is proposed that tries to achieve the ideal target rates.

In particular, we would like to address the following issues : a) Can we reduce the impact of contract rates on the difference between realized rates and target rates b) Can we contain the non-responsive UDP sources ? and c) Can we reduce the impact of RTT-bias on realized throughput?

The rest of the thesis is organized as follows. In chapter II, we propose an approach to active resource management in differentiated service network. We call this mechanism TRIO (Three colored localized marker using RIO). In chapter III, the implementation details of TRIO are given. Chapter IV presents the simulation results of TRIO's performance under various conditions and chapter V discusses related work. Chapter VI concludes the thesis.

CHAPTER II

TRIO - A PROPOSAL FOR ACTIVE RESOURCE MANAGEMENT

Recent studies and simulations show that when the resources are plentiful, the realized rate is not proportional to the reserved rate [13, 15]. When the links have little or no excess bandwidth, realized rates are closer to the target rates. This is the key concept which is used in the development of TRIO. This means that if we can virtually make the link loaded , we can realize the throughput closer to the target rates and hence achieve the desired service differentiation.

How to make the link virtually loaded? A link is said to be virtually loaded when it has no apparent excess bandwidth. Every router will know the capacity of its output link C. If it knows the amount of IN traffic going through it, then it can find how much of excess bandwidth E it has. Then all the router has to do is to convert the OUT packets at the rate of E into IN packets. That way, the router will be operating as if the reserved rate equals the capacity of its output link. Since the capacity of this link wouldnot be equal to that of another router down the network path, it cannot assume that this conversion of $OUT \rightarrow IN$ is valid for the other router. So it has to rename the packets which it converted to IN as OUT before it puts them on its output link. For this purpose we identify those packets that are converted from $OUT \rightarrow IN$ as IN2 packets. As shown in figure 6 just before dequeuing, we convert those packets with IN2 as OUT.

The changes in packet markings are localized and hence no standardization across the network is needed. This makes this scheme easily deployable.

Now the buffer management of TRIO will need to identify 3 levels of packets. IN/IN2/OUT. It is graphically shown in figure 7. The new IN2 marked packets have a precedence between IN and OUT packets. The threshhold for OUT packet are

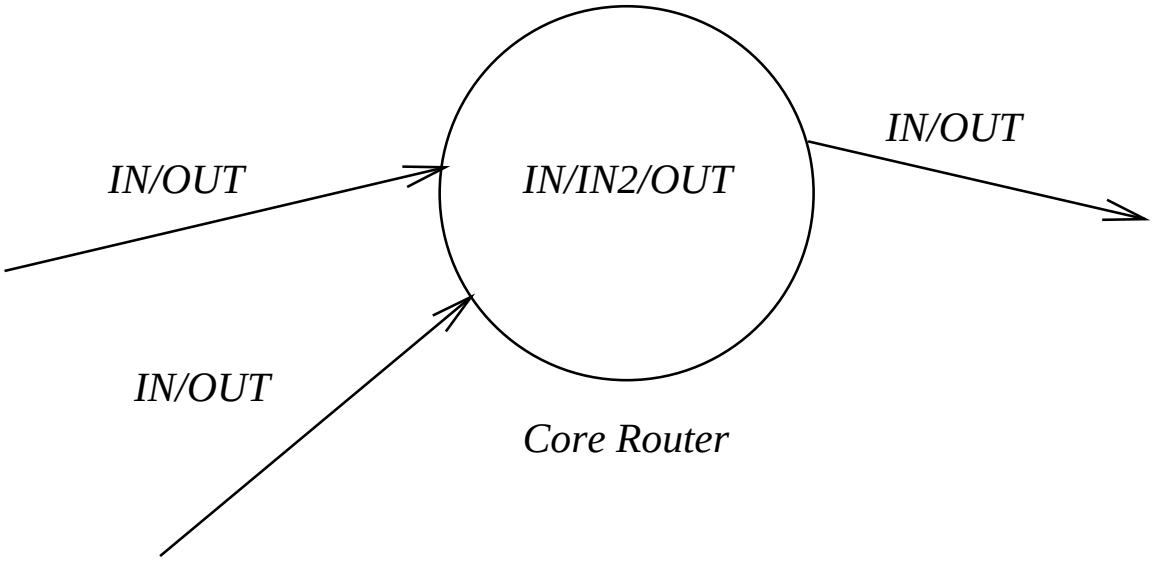

Fig. 6. 2-levels in the network and 3-levels in the core router

moved further down. As a result, very few OUT packets, that are not converted into IN2, get through the routers.

In the rest of this section, we will discuss how to identify the excess bandwidth and on what basis we can convert $OUT \rightarrow IN2$. By sampling, the router can find the rate of the IN traffic without maintaining per-flow state information. Hence it can easily estimate rate at which the OUT packets need to get converted into IN2. But the important question is which OUT packets should be converted into IN2? It is not advisable to convert , indiscriminately, all the OUT packets to IN2 at the rate of the excess bandwidth. For, such probabilistic marking still allows a large number of packets of an irresponsive flow to pass through the router.

If we maintain state information for each flow, we could calculate the OUT rate of every flow OUT_i. Then, we can calculate the ideal dropping rate for this flow as 1 - α/OUT_i. Such a strategy is adopted by [17] where edge routers measure the flow rates

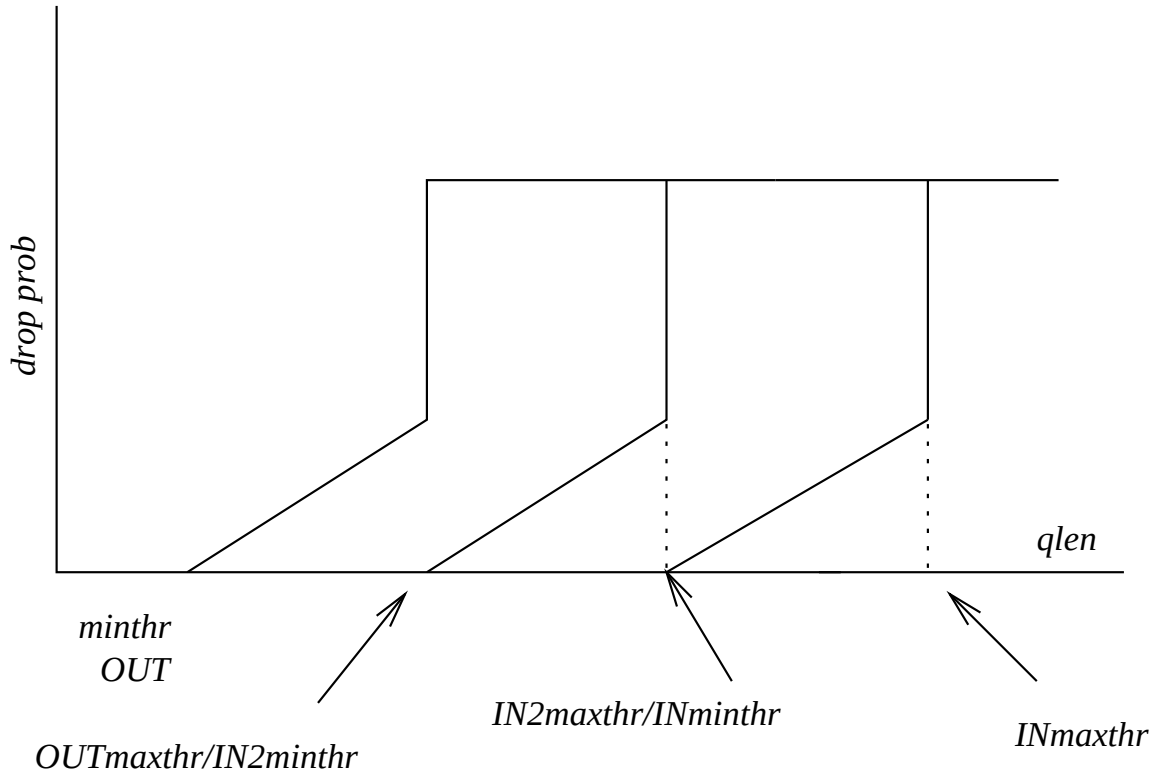

Fig. 7. TRIO with IN/IN2/OUT

and this state information is carried as a part of the packet header. This mechanism requires that routers be standardized to mark the packet state uniformly. Diff-serv architecture already uses IP packet TOS byte to mark the packets differently and hence the approach in [17] is not directly compatible.

A different approach is taken in this thesis. We propose an approach which maintains a partial state information at each core-router. The state carries information about some of the flows that are exceeding the fair share. For these flows, for which the state is maintained, we can accurately curtail their OUT_i bandwidth to fair share α. For other flows, for which the state information is not maintained, we aggregate the OUT_i bandwidth and allow buffer management to contain them with in the excess bandwidth.

This approach is similar to how caches are employed in modern computer systems to reduce average memory latency. This engineering solution can reduce average memory access time considerably if a high fraction of the data are found in cache. Similarly, we plan to use partial state to cache state for a few flows that are sending at a high rate. By actively managing the resource usage of these flows, it is expected that we may be able to improve the overall fairness of bandwidth allocation. In [18], the authors show that this can improve fairness in sharing link bandwidth. Our work here targets the specific problems of diff-serv networks and the solution we propose here is much simpler and different from [18]. Caching has been used for router-lookup and other functions in routers [19]. Sampling has been employed for network management in [20].

TRIO doesnot explicitly drop the OUT packets that are not converted into IN2. We let the buffer management scheme control the rate at which these packets get dropped. As shown in figure 7, the OUT packets are set to have low queue thresholds and hence will be dropped at a higher rate whenever the queues build up. It

is emphasized that TRIO actively allocates excess bandwidth so that IN2 marked packets get through the router at a higher probability than otherwise.

Schemes that try to increase the drop rate of OUT packets of non-responsive flows, without marking the remaining packets differently, proved ineffective. In such schemes that do not employ different levels of markings, when queue lengths are below minimum threshhold, no packets - neither the packets of responsive flows nor those of non-responsive flows, are dropped. When queue lengths are above the threshhold, packets of non-responsive flows are dropped and so are the packets of responsive flows , albeit at a lower rate. As a result of packet drops and subsequent reduction of sending rates of responsive flows, the queue lengths remained below the threshhold most of the time. As a result, non-responsive flows escaped packet drops making such schemes ineffective.

The scheme that is explained in detail in the next section employs the above ideas; (a) When IN bandwidth is close to link capacity, realized rates are close to target rates; hence in all situations, virtually minimize the excess bandwidth at each router and (b) employ partial state at each router to properly manage the excess bandwidth (c) employ localized packet markings to distinguish between OUT packets of different flows.

Although this method which is employed in this thesis requires the core router to recognize 3 different kinds of packets IN/IN2/OUT, it is not similar to the three-drop precedence mentioned in [9]. Where as three-drop precedence requires the R/Y/G to be marked at edge routers depending upon a pre-determined R/Y/G levels, the method that is used in this thesis converts $OUT \rightarrow IN2$ at every core router based on the excess bandwidth on that particular output link.

Continuing with the discussion of which OUT are to be converted into IN2, this is the place where the sampling and caching part of TRIO comes into picture. There

are limited number of entries in the cache. The flows are sampled and cached and are monitored to see whether the rate of their OUT packets is more than their share of the excess bandwidth. If that is the case, they are individualized. In section 3, we will find how exactly the flows, whose OUT rate exceeds the fair share of the excess bandwidth, are individualized and the others are aggregated.

Is TRIO scalable i.e., if we increase the number of cache entries, will we get better performance? If there are more non-responding flows than the number of cache entries, will they be caught in the next core router with TRIO i.e., Does TRIO possess the additive property? Since the network is dynamic, how correctly can we predict the excess bandwidth? How is the performance of TRIO affected if excess bandwidth is mis-calculated? In section 4, through simulations we show that TRIO is scalable and that it possesses the additive property.

Simulation studies have been done to use a partial state information for realizing fair sharing of bandwidth in the exisiting internet infrastructure [18]. A non-responding flow is identified by caching and monitoring over an observation period. The advantage of the mechanisms that use partial state is that the searching is done in O(1). It is shown by the author that a non-responding flow is identified with a probability of 0.95 in 25 observation periods with state to monitor 10% of the total number of flows.

In the next chapter we discuss the implementation details of TRIO.

CHAPTER III

IMPLEMENTATION OF TRIO

TRIO recognizes 3 levels of packet markings IN/IN2/OUT when the edge router employs IN/OUT markings. TRIO implementation with three color edge marking [9, 10] is discussed later. TRIO consists of 2 main parts. The first part identifies the flows whose OUT rate (OUT_i) exceeds their share of excess bandwidth. This is done through sampling and caching. Flows are sampled and monitored to see if their OUT_i exceeds α. If it is observed that a flow's outrate is greater than α, that flow is pinned. State information, OUT_i, is maintained for all cached flows. The OUT packets of pinned flows are converted into IN2 at the rate of α/OUT_i. The OUT packets of non-pinned flows are converted into IN2 at an aggregated rate as described below.

The second part takes care of dropping of packets at the time of congestion. This is done through active buffer management. TRIO adjusts the queue thresholds appropriately such that IN/IN2 marked packets get through the router and very few packets marked OUT (not converted into IN2) get through the router.

In this section, we describe the details of the first part. The flowchart shown in figure 8 depicts the different modules and the overall flow of TRIO. The following description of TRIO's operation closely follows the flowchart in figure 8.

If the arriving packet is marked IN, a counter for estimating the total IN rate is incremented and the packet is enqueued for the output link. If the arriving packet is marked OUT, the cache is checked to see if the packet belongs to a cached flow. If the packet doesn't belong to a cached flow, it is checked to see if this flow can be cached. If there is room in the cache, this flow is cached and the out packet counter for this flow is initialized.

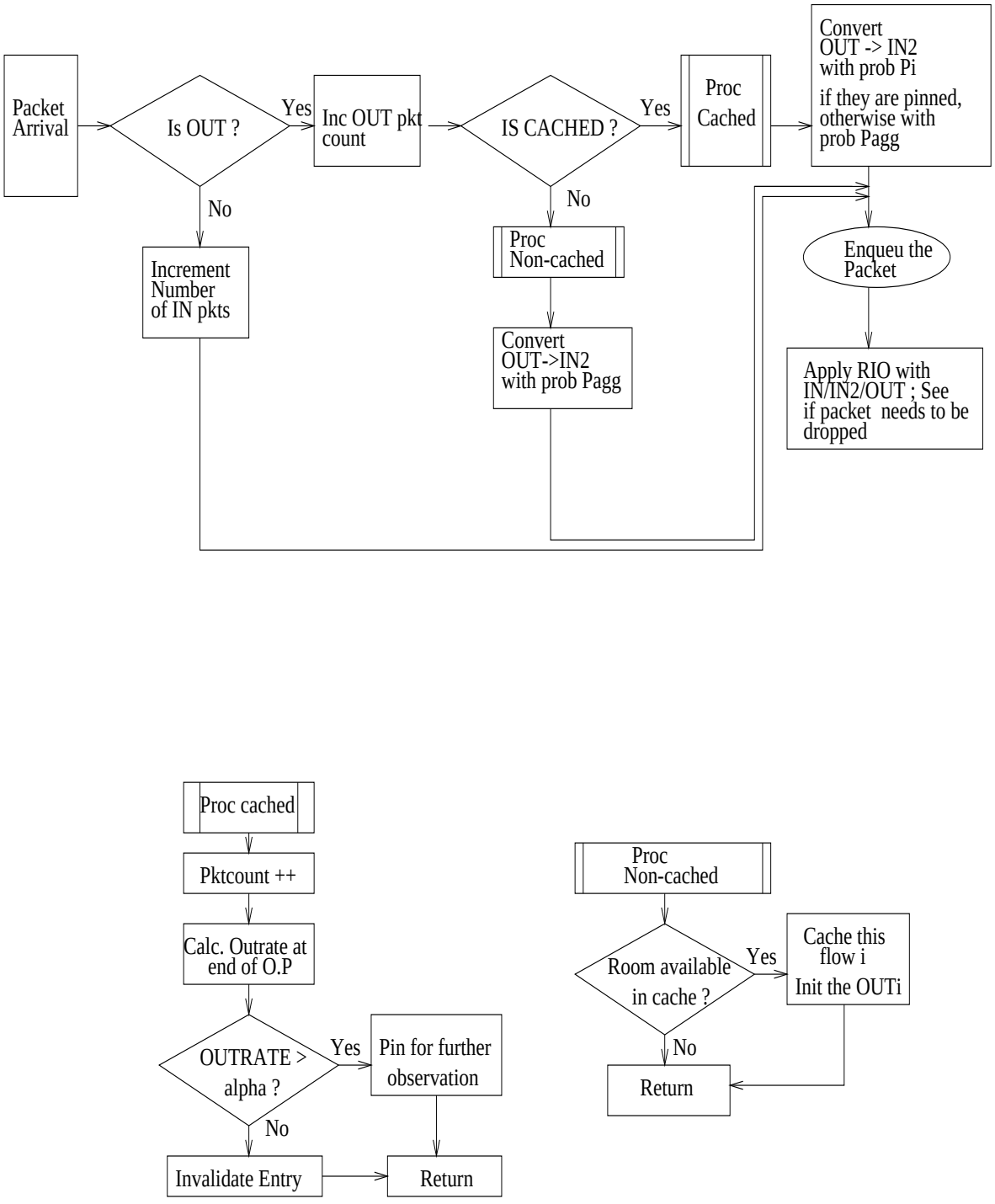

Fig. 8. Flow chart describing the operation of TRIO

If the arriving OUT packet belongs to a cached flow, the packet count is incremented for this flow. If the observation period for this flow has expired, the out rate is calculated by dividing the packet count by the observation period. If the resulting outrate is below α, this cache entry is marked invalid making room for another flow. If the out rate is above α, this entry is *pinned*, and the OUT packet of this flow is converted to IN2 with a probability of α/OUT_i, where OUT_i is the outrate observed in the last observation period.

If the arriving OUT packet belongs to a non-pinned flow, it is converted into IN2 with a probability of p_{agg}. The fairshare of the excess bandwidth, $\alpha = E/N = (C - IN)/N$, where N is the number of flows going through the router. Let OUT_{total} be the total out rate, OUT_{pinned} be the out rate of the *pinned* flows in cache and l be the number of flows that are pinned in the cache. The probability that an OUT packet of a non-pinned flow is converted from OUT to IN2 is

$$p_{agg} = (C - IN - \alpha * l)/(OUT_{total} - OUT_{pinned}) \tag{3.1}$$

In the above equations, IN, OUT_{total} and OUT_{pinned} can be easily measured through counters. The only challenge is how to estimate the number of flows, N, going through the router. Determining the number of flows accurately is a difficult task because we don't want to keep per-flow state information. We estimate the number of flows as accurately as possible. In [21], the authors have approached this problem by randomly sampling the incoming packets to see if they get a hit against a cache full of flows. The number of flows then can be estimated based on this cache hit ratio and the number of entries in the cache. If all the flows send packets at an equal rate, $p_{hit} = K/N$, where K is the number of entries in the cache. By measuring p_{hit}, we can then calculate N. However, in the general case, not all the flows are sending packets at the same rate. Moreover, TRIO tries to cache the flows that are sending

packets at a higher rate than others. Hence, observed p_{hit} will most likely be higher than K/N. If we calculate $N = K/p_{hit}$, we will be underestimating the number of flows. As a result, we will overestimate α. We call this estimate of number of flows as hieuristic 1. If the OUT packets are converted into IN2 based on α/OUT_i, since α is overestimated, we will be converting a higher fraction of packets of the non-responsive flows into IN2. This is undesirable.

In our implementation, flows observed to exceed the fair share α are *pinned*. These are the flows with high rate of OUT packets. When calculating p_{hit} empirically, if we consider these pinned flows also, we will under estimate the p_{hit} and hence will overestimate the number of flows. We call this estimate as $N^{'}$ and hieuristic 2 in the rest of the thesis. Now we can hieuristically place the actual number of flows, say $N^{''}$, in between N and $N^{'}$. In this work, $N^{''}$ is calculated as an average of N and $N^{'}$. We call this average estimate as hieuristic 3. We will show that results are pretty close to the ideal rates with either of these estimates. We will also present results to show the impact of misestimation of α.

A. Cost analysis of TRIO

As mentioned earlier, there are 2 main parts in TRIO viz., (i) identifying a non-cached flow plus calculation of $OUT \rightarrow IN2$ probability (ii) dropping packets if needed according to the OUT/IN2/IN thresholds.

The IN packets will go through only the 2nd part. Please refer to figure 8. The additional cost in TRIO for an IN packet is the cost of updating the IN packet counter.

The OUT packet will go through identification and probability calculation phases. In the identification phase, cache insertion , searching and deletion are involved. Depending upon the associativity of the cache, the search time will vary. Higher the

associativity, higher the time for search without parallel hardware assistance. Cache lookup can be done with O(1) time. Also at the end of an observation period, the *outrate* of a flow is calculated. As shown in previos section, the *outrate* involves division. But by choosing suitable observation periods (for example, a power of 2), this division can be done using simple shift operations. In addition to these above mentioned calculations, we need to keep track of number of OUT packets, IN2 conversion rate for non-cached flows and the cache hit rate. Probability calculations are done only at the end of an observation period. All of these calculations are O(1) per packet arrival. Work done per cached flow is higher than non-cached flows and as more processing (and memory) resources become available, larger caches can be employed.

B. TRIO with 3-color edge marker

The 3-color edge marker is discussed in [9]. In this method, the edge router differentiates the packets as RED(R), YELLOW(Y) or GREEN(G) based on the predetermined thresholds for R,Y,G. With the 3-color edge marker, TRIO cannot convert $R/Y/G \rightarrow OUT/IN2/IN$ directly. We propose the following method to integrate 3-color edge marker with TRIO.

At each router, we would like to convert as many yellow packets as possible to IN2 before any red packets are converted into IN2. This brings up two possible cases. Let G, Y, and R denote the total rate of Green, Yellow and Red packets respectively at a router. If $Y > C - G$, then do not convert any $R \rightarrow IN2_R$ and convert $Y \rightarrow IN2_Y$ with a probability,

$$P_{Y \rightarrow IN2_Y} = (C - G)/Y \tag{3.2}$$

If $Y < C - G$, then we convert all $Y \rightarrow IN2_Y$. The probability for $R \rightarrow IN2_R$ depends upon whether a flow is cached or not. If the packet belongs to a flow i that is pinned in the cache, the probability of conversion to IN2 is given by:

$$P_{R_i \rightarrow IN2_R} = (C - G - Y)/R_i \tag{3.3}$$

If the packet belongs to a flow which is not pinned and let $R_{agg} = R$ - R_{pinned} , then the probability that R is converted into IN2 is given by

$$P_{R_{agg} \rightarrow IN2_R} = (C - G - Y)/R_{agg} \tag{3.4}$$

Since we have to put back the IN2s into R and Y on the output link, we convert the $Y \rightarrow IN2_Y$ and the $R \rightarrow IN2_R$. This allows us to preserve the original markings of Y and R as the packets exit the router. The above equations show that for the cached flows, we only need to maintain state for counting the Red packets. Only aggregate states for Y and G are necessary. Since a flow cannot send unlimited *yellow* packets, individual flow state is not needed when $Y > C - G$, as observed in equation 3.2.

Figure 9 shows the drop probability for different categories of flows in the above scenario. Please find that two different threshholds for $IN2_R$ and $IN2_Y$ donot exist. Since the differentiation is only for the purpose of reverting to R or Y on the output link, the two threshholds are coupled into one called the IN2 threshhold.

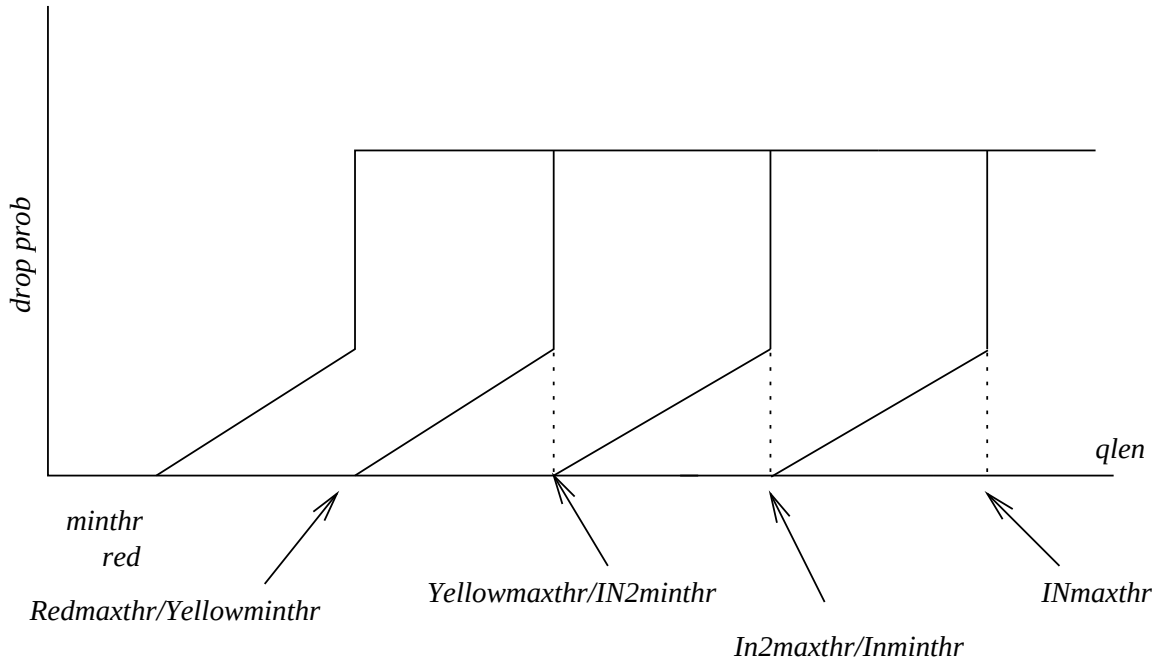

Fig. 9. TRIO with RYG

CHAPTER IV

SIMULATION RESULTS FOR TRIO

In this chapter we present the results of the simulations under different conditions for TRIO. The common setup is as shown in figure 10. There are totally 40 flows out of which 8 are UDP and the rest are TCP-Reno. The UDP flows do not respond to congestion. In the EdgeRouter R1, we have the marker installed and in the router R2, we have TRIO running. The RTT for all the flows are 8ms and the individual link to the router has 10Mb capacity. TRIO is configured with 0/0/500/600 for the minthreshhold for OUT, maxthreshhold for OUT, maxthreshhold for IN2, maxthreshhold for IN respectively. The observation period is 20ms. Unless otherwise mentioned, flows 1-8 have 0Mbs reservation, flows 9-16 have 0.1Mbs reservation, flows 17-24 have 0.5 Mbs reservation, flows 25-32 have 1Mbs reservation and flows 33-40 have 2Mbs reservation.

A. Containment of non-responsive flows

We employ non-responding UDP flows as example non-responsive sources. In this experiment, the UDP flows have no reservation and hence should receive only best-effort service. The bandwidth in the link R1-R2 is 115.2Mbps, and the contracted rates of other flows amount to a 25 % reservation on the link. Figure 11 is plotted for the plain RIO and TRIO with 12.5%, 25%, 100% state information.

It is evident from the graph that TRIO has contained non-responsive UDP flows and has distributed the excess bandwidth equally among all the flows. RIO fails to contain these UDP flows and hence they claim a significant amount of the link bandwidth while responsive TCP flows fail to get their fair share of the bandwidth. It is also observed that a 3-color edge marker doesn't offer significantly better per-

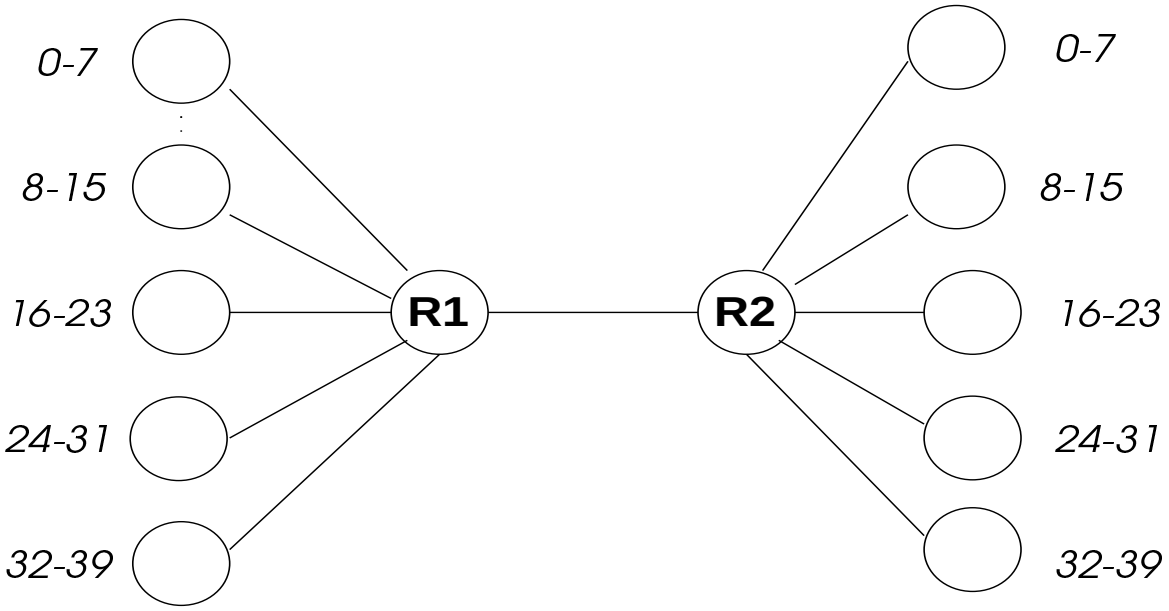

Fig. 10. Simulation setup

formance. When the state information exceeds the number of UDP flows, they are all cached and contained to get only the fair share of excess bandwidth. When state information is not sufficient, some UDP flows escape observation and hence obtain larger than their fair share of bandwidth. With enough state information, all the flows achieve throughput closer to their target rates.

In figure 12, we show how TRIO performs when the link capacity is 115.2Mb, 57.6Mb, 38.4Mb and 32Mb i.e., when respectively 25%, 50%, 75%, 90% of the total link capacity is reserved. It is observed that in all the cases, TRIO contains UDP flows to their fair share irrespective of the reserved load on the link.

We conducted another experiment to test the efficacy of containing non-responsive flows. In this simulation, the UDP flows have a contract rate of 0.5Mb. The results shown in figure 13 demonstrate that TRIO does give the UDP flows their share of reserved bandwidth and does not blindly punish the non-responsive flows.

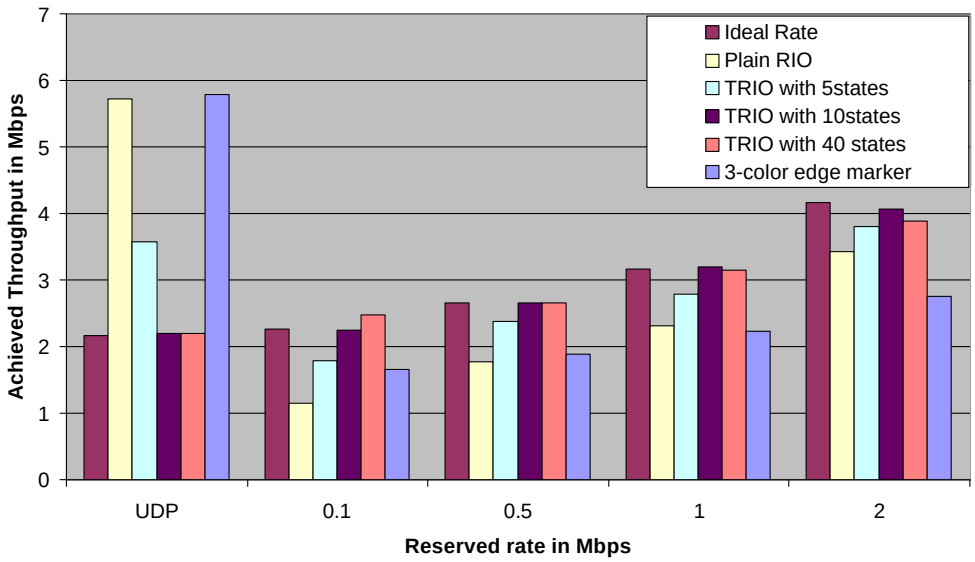

Fig. 11. TRIO performance with UDP reservation 0Mbps

Also figure 13 plots the results for the simulation with queue threshholds of 10/80/500/600. It is observed that the results are nearly the same as with queue thresholds of 0/0/500/600. Our experiments have shown that as long as the IN2 packets have significantly higher queue thresholds, they are protected from the OUT packets of non-responsive flows.

These results indicate that TRIO is effective in identifying non-responsive sources and at limiting their bandwidth consumption to their fair share. It is also observed that the realized rates are close to target rates irrespective of the reservations or

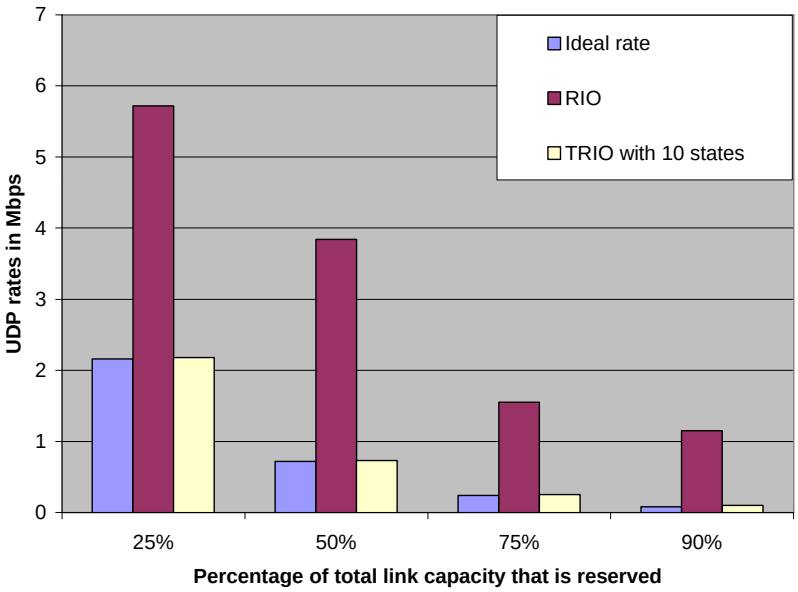

Fig. 12. UDP containment with different link capacity reserved

contract rates.

B. Reducing RTT bias

To observe TRIO's efficacy at reducing the RTT bias, we conducted simulations with TCP flows with different RTTs. The RTTs of TCP flows within each reservation are varied from 8ms-36ms. Let *diffmaxmin* be the difference between max. and min. throughput of the flows in the group with the same reservation. Figure 14 plots the ratio of *diffmaxmin* to the target rates of the flows. It can be clearly seen from the

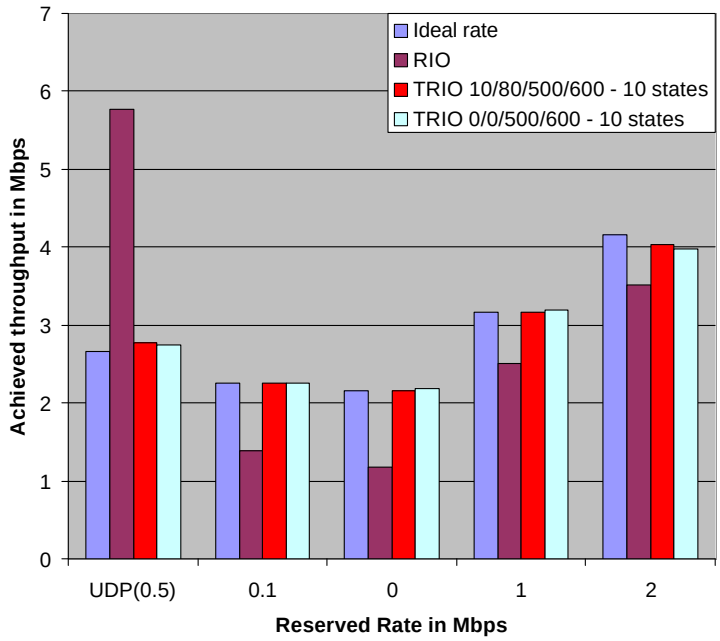

Fig. 13. UDP Containment with UDP reservation 0.5Mbps

figure that TRIO reduces RTT bias compared to plain RIO.

C. Impact of excess bandwidth estimates

Figure 15 compares the performance of TRIO when we know the number of flows, N, that are going through the router with that of TRIO when we estimate N. Estimate 1 and Estimate 2 in figure 15 refer to N(hieuristic 1) and N''(hieuristic 3) discussed in chapter III. From the figure 15, we find that performance with N'' estimate is close to the ideal case of the number of flows being known. It is observed that the

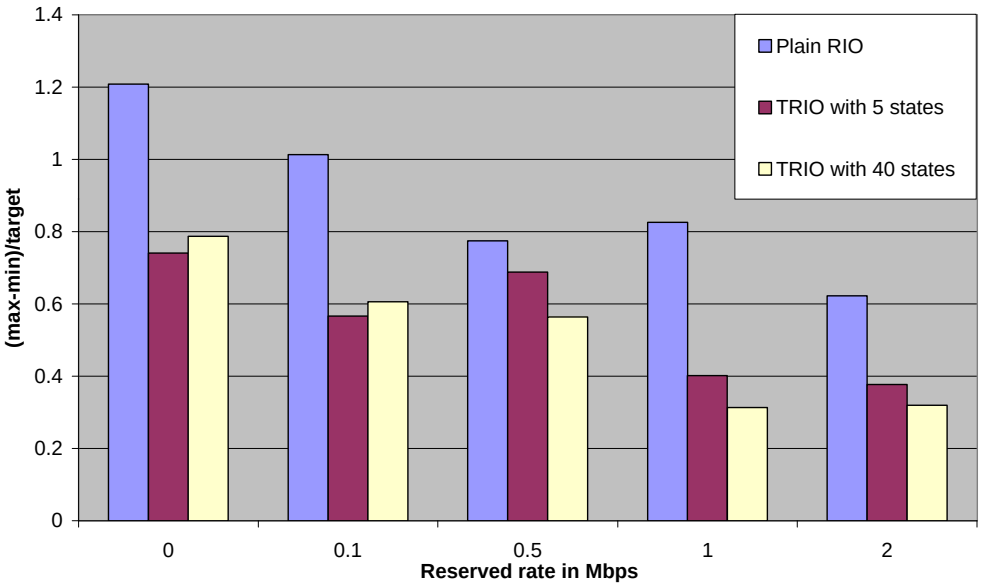

Fig. 14. TRIO with TCP flows of different RTTs

performance with N estimate is slightly worse. In all cases, TRIO's performance is much better than that of RIO.

To study the impact of miscalculation of excess bandwidth further, we conducted another experiment. In this simulation, the fair share excess bandwidth parameter α is artificially set either too high or too low. In one case, we set it to 200% of the actual value of α and in another case, we set it to 50% of the actual value of α. This is expected to illustrate the potential range of TRIO performance. From figure 16, it is observed that even when we over-estimate α by 100%, obtained results are

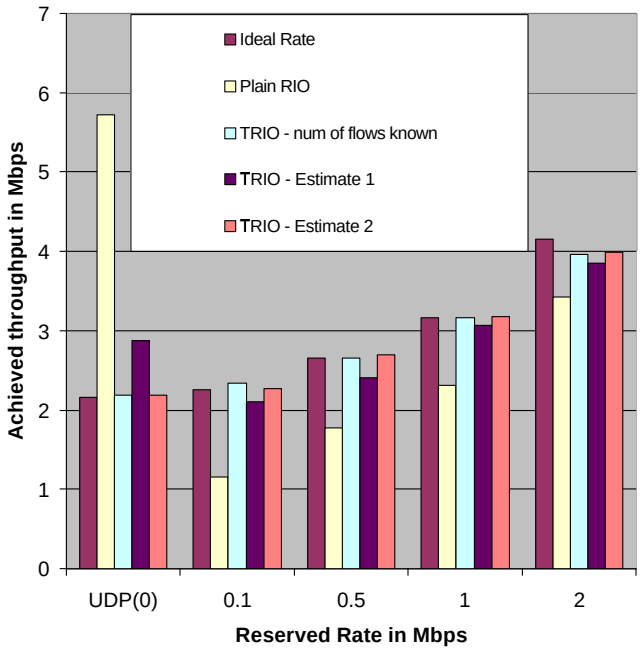

Fig. 15. Performance of TRIO with different estimates of flows

better than RIO's results. Underestimation of α leads to aggressive containment (or punishment) of non-responsive sources.

D. Impact of multiple TRIO routers

Earlier experiments have shown that TRIO can be effective in improving the provided bandwidth service at a single router. How well does performance improve as more routers in the network deploy TRIO? We also wanted to see if multiple routers can work together to contain non-responsive sources when the state at a single router

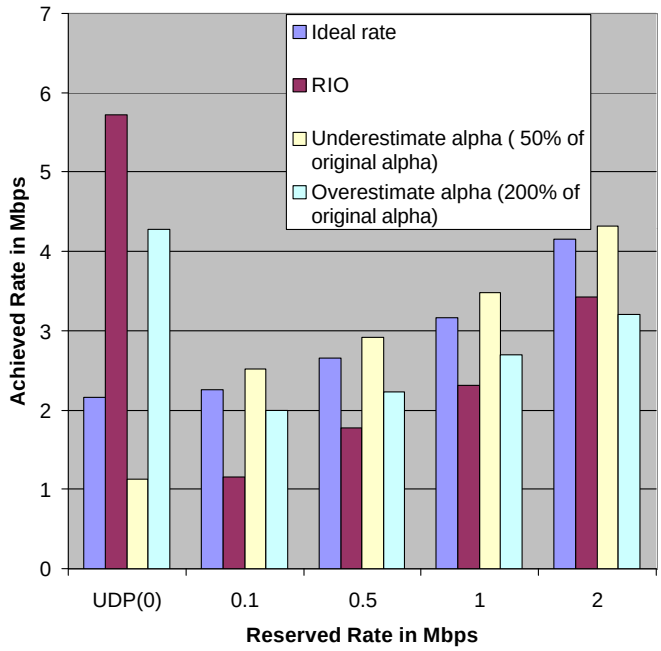

Fig. 16. Impact of overestimating or underestimating alpha

is not enough to cache all the non-responsive flows. In this experiment we setup 3 routers, so that routers R2, R3 run TRIO. The setup is as shown in figure 17. The link R1-R2 as well as R2-R3 have 25% of their link bandwidth reserved.

We have just 5 states in the cache of R2 while there are 8 non-responsive flows. We observe from the simulation results that the 3 UDP flows that escape R2 get caught in R3. This is shown in the figure 18. This shows that as more TRIO routers are employed, the provided service is improved. This shows that the state of all the routers can be combined and employed to contain non-responsive sources.

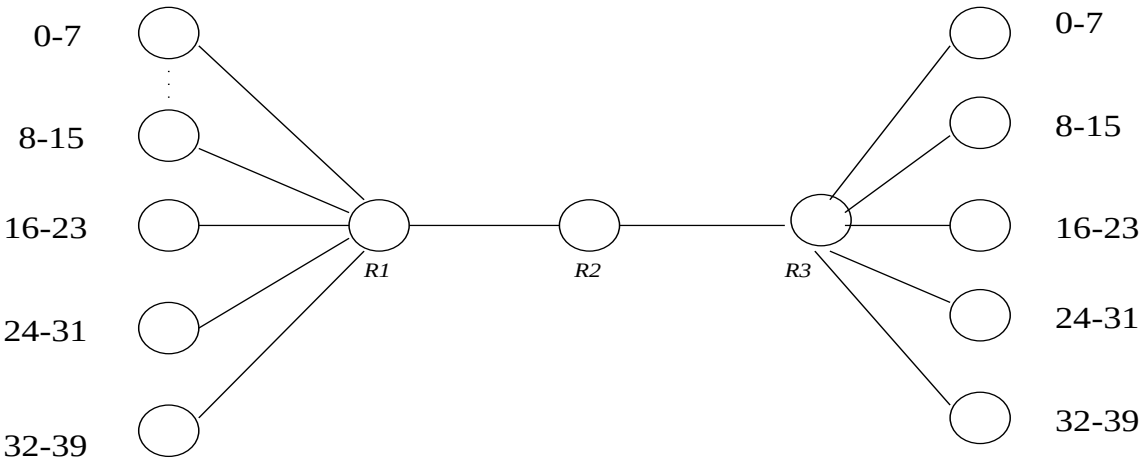

Fig. 17. The 3 router setup

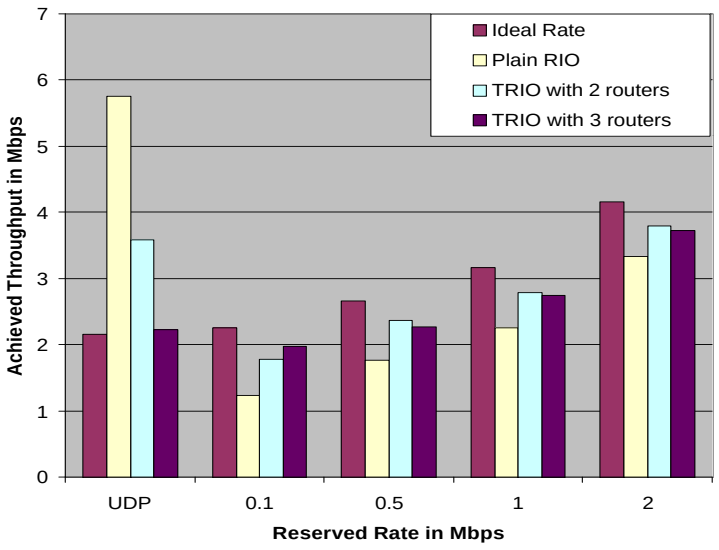

Fig. 18. Impact of multiple TRIO routers

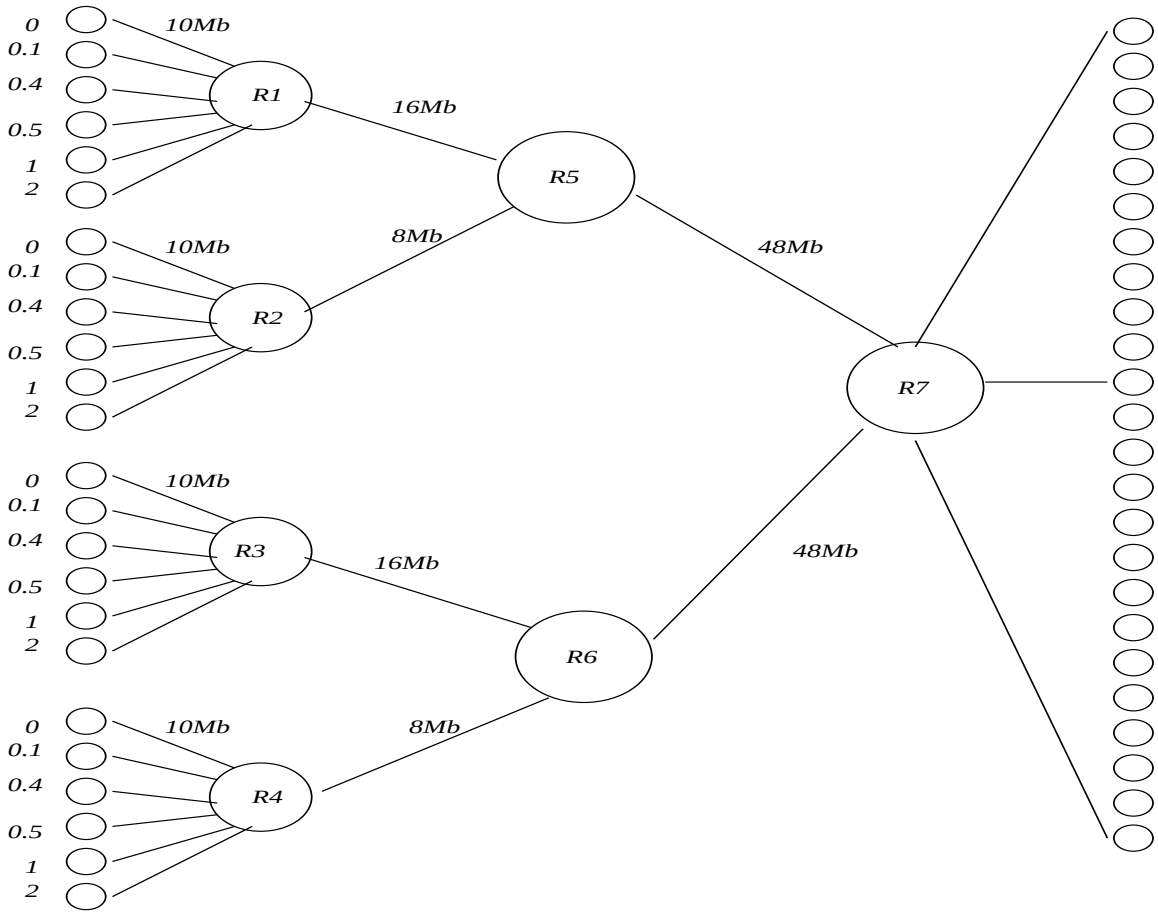

Fig. 19. Simulation topology in a heterogenous network

To observe the performance of TRIO in more detail, we did simulations in a complex network topology with multiple routers. The setup is as shown in figure 19.

The results are shown in 20. The results clearly show that TRIO performs better than RIO by containing the UDP flows and by realizing throughputs close to the ideal target rates.

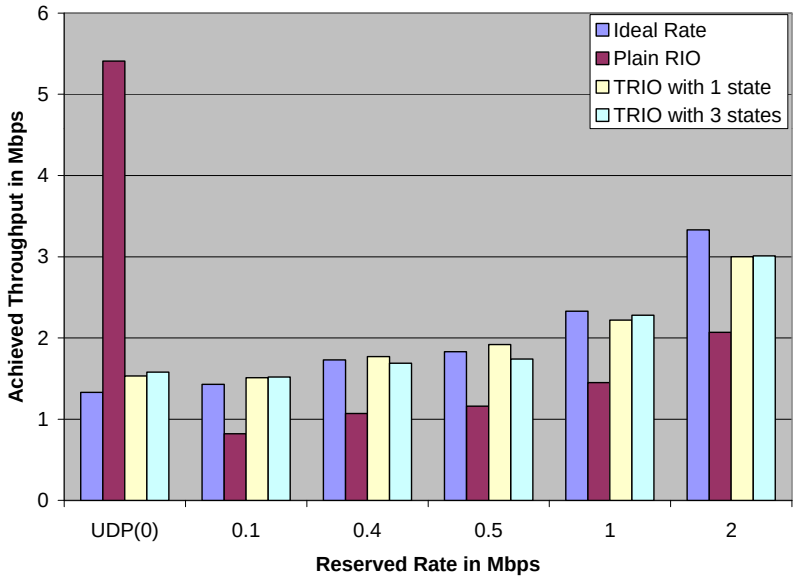

Fig. 20. Performance of TRIO in complex topology

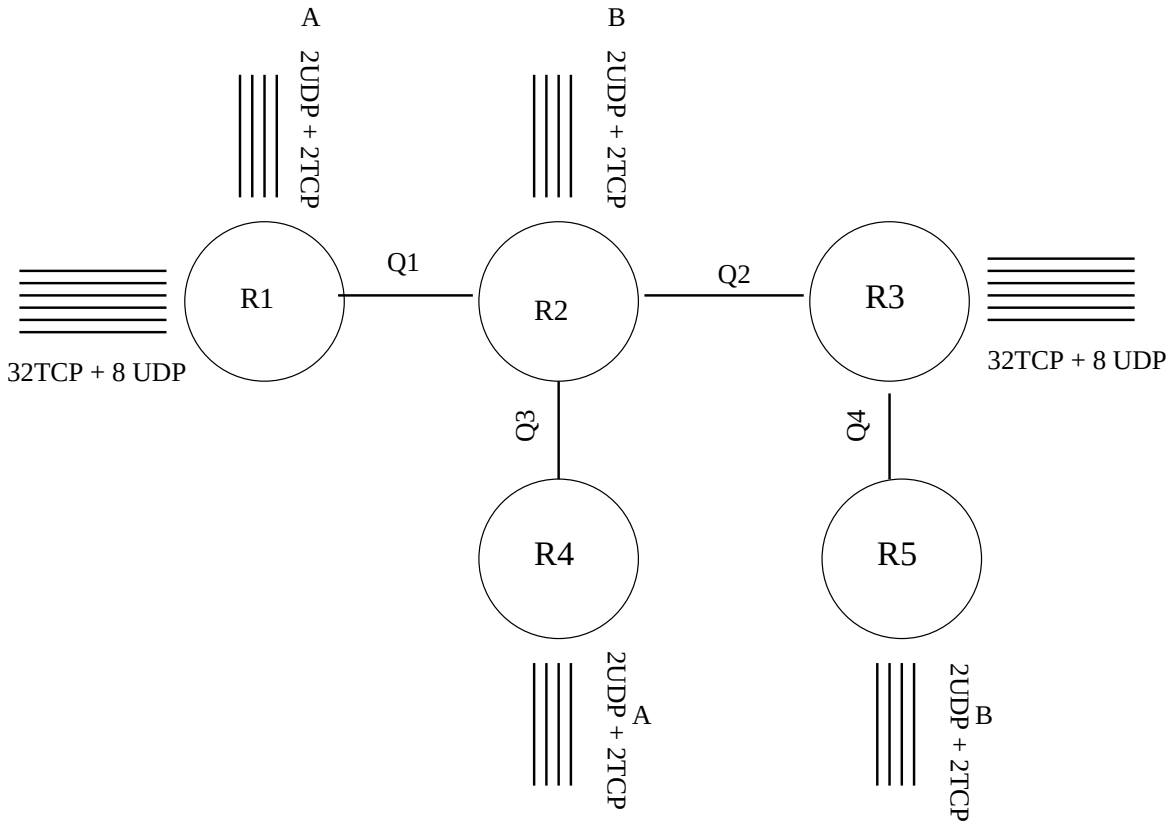

Fig. 21. Cross-traffic simulation setup

E. TRIO with a cross-traffic network

The following simulation is done with cross-traffic flowing across the network. The topology is as shown in figure 21. There are two cross traffic sources named A and B. Both have 2UDP flows and 2TCP flows. The UDP flows have 0Mb reservation and the TCP flows have reservation of 2Mbps. The link bandwidth are setup so that all of them have 25% of their bandwidth reserved for IN traffic. Q1 has state for maintaining information about 7 flows where the rest of the queues maintains 5-flow state information.

10 UDP flows go through Q1 . Since the 3 escaped flows get caught in Q2, the

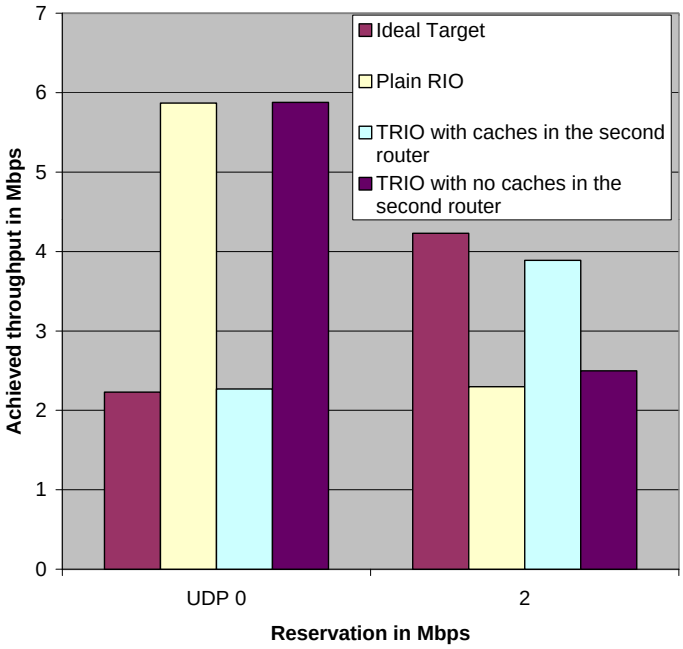

Fig. 22. Cross-traffic results

TCP performance of the crosstraffic B will be benefitted if TRIO is implemented in Q2. The figure 22 shows the performance of cross-traffic B with Q2 having TRIO and without TRIO.

The results show that TCP flows of cross-traffic B are indeed benefitted by the additive property of TRIO.

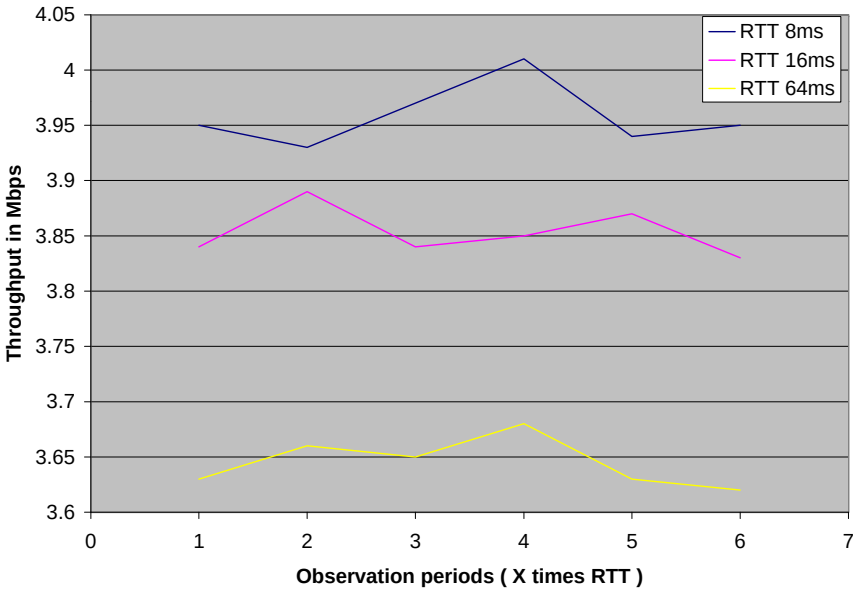

Fig. 23. Effect of observation periods

F. Effect of observation periods

A set of simulations are done to find the effect of observation period on the realization of throughput. With a constant RTT across all the flows, the observation period is varied from 1 RTT to 6 times the RTT. This simulation is repeated by varying the RTT. The results are shown in figure 23.

We can observe that the difference in throughput due to changing observation period is small compared to that across different RTTs.

G. Estimate of number of flows by hieuristics

The following simulation is done to find the efficiency of estimating number of flows by hieuristics mentioned in chapter IV. The simulations are done for 75 seconds and the flows dynamically start and stop as given in the table I . An X means that that particular flow is present during the interval mentioned in the header .

Table I. Dynamic flows setup

Time	0-15	15-30	30-45	45-60	60-75
UDP	X	-	-	X	X
TCP 1-16	X	X	X	X	-
TCP 16-24	X	X	X	X	-
TCP 24-32	X	X	-	-	-
Total	40	32	24	32	24

For this setup, the time taken to reach an estimate with error margin $\pm 10\%$, the percentage of time spent within an error margin $\pm 10\%$, that spent within an error margin $\pm 20\%$ are given in the table II.

Table II. Performance of estimation of number of flows

Time	0-15	15-30	30-45	45-60	60-75
time in secs. taken to reach an estimate with error margin ±10%	3.187	1.9	1.5	1.02	2.85
Time spent within ±10%	70%	40%	47%	74%	50%
Time spent within ±20%	86%	75%	68%	100%	90%

The dynamic estimation of number of flows along time for the above setup is shown in figure 24. The "under-estimated" series in the graph relates to hieuristic 1, "over-estimated" to hieuristic 2 and "average" to hieuristic 3 discussed in chapter III.

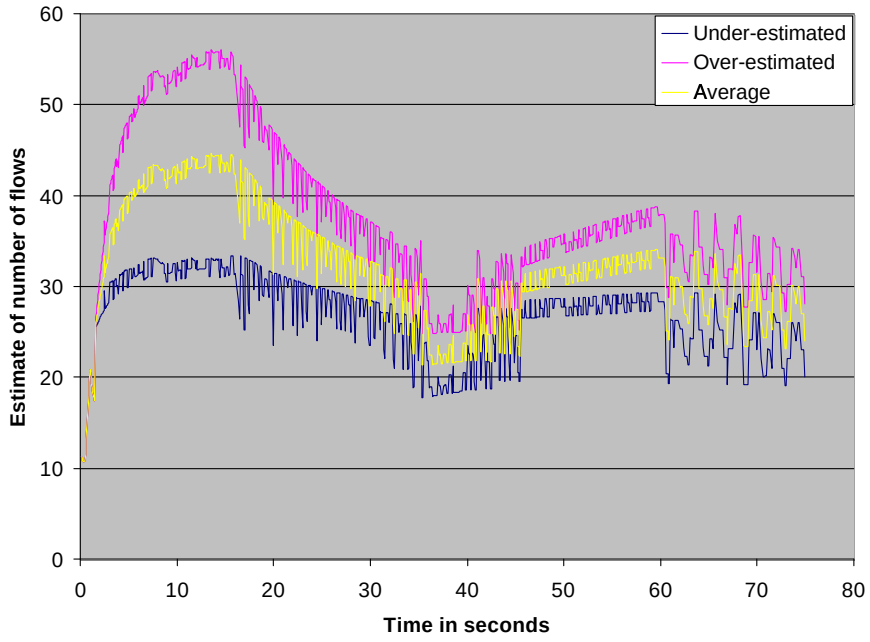

Fig. 24. Dynamic estimation of number of flows

CHAPTER V

RELATED WORK

A number of approaches have been recently proposed to improve the performance of assured forwarding PHB of diff-serv architecture. These include improved markers [9, 10], contract-rate based droppers [11], and modified TCP sources [22]. Each of these schemes have their strengths and weaknesses. Three color markers are not found to be universally effective [23]. Contract-rate based droppers require that the packets be marked with their contract rates and hence don't immediately fit into diff-serv framework [6]. Deploying modified TCP sources will likely to be a significant effort. TRIO is shown to be effective in improving the performance while requiring changes only to the diff-serv networking elements.

Class based queuing can be used to contain the effects of UDP flows on TCP flows. This however penalizes all the UDP flows whether they are responsive or non-responsive [24]. Per-flow RED or FRED [25] can be used to contain non-responsive sources while achieving fair sharing of bandwidth. FRED however requires 100% state. TCP-friendly test [14] is suggested as a mechanism to contain non-responsive flows. This approach requires RTT measurements of flows and also requires that all the flows follow a TCP-style exponential decrease of rates when a packet is dropped.

In [17], edge routers measure the flow rates and this state information is carried as a part of the packet header. This packet state is used at a router to allocate the bandwidth fairly among the flows. This mechanism requires that routers be standardized to mark the packet state uniformly. Diff-serv architecture already uses IP packet TOS byte to mark the packets differently and hence the approach in [17] is not directly compatible. TRIO also relies on the estimation of a flow's fair share of the bandwidth as the scheme in [17]. However, these schemes are considerably different:

(a) TRIO calculates fair share of bandwidth using cache hit rates and packet counters, and (b) fair share of bandwidth is used to calculate packet drop probabilites in [17] and is used here to calculate packet remarking probabilities; packet-drop decision is made by the buffer management policy in TRIO.

Previous approaches [14, 17, 18] have relied on calculating drop rates for containing non-responsive flows. TRIO relies on resource allocation and allows packet-drop decisions to be made by the buffer management policy. This results in greater flexibility in the management of non-responsive flows.

Diff-serv architecture allows packets to be remarked within the network, for example, as packets leave one service provider's domain and enter another service provider's domain [6]. However, such remarkings stay visible to all the routers in the second domain. In TRIO, the packet remarkings are localized to a single router. As they leave a router, the packets are marked exactly the same way as they arrived at that router.

Caching has been employed in networking elements in various forms, for example, during route-lookup. Caching has been recently proposed for maintaining partial sate within network elements [18]. Our work employs similar ideas, but the solution proposed here is specific to diff-serv architecture and is considerably simpler. Caching has also recently been employed for estimating the number of flows passing through a router [21].

CHAPTER VI

CONCLUSION

In this thesis, a novel approach has been proposed for active resource management in a differentiated services network. The proposed approach, TRIO, employed: (a) active resource allocation through localized remarking of packets at a router, (b) caching to identify flows sending OUT packets at a high rate, and (c) active buffer management for realizing ideal target rates. It was shown that TRIO (a) can be very effective in containing non-responsive flows, (b) can reduce the impact of contract rates on the difference between realized rates and target rates and (c) can reduce the RTT bias among responsive TCP flows. It was also shown that service is improved as more routers deploy TRIO. It was shown that the packet handling cost remains O(1) in TRIO.

REFERENCES

[1] A.Demers, S.Keshav, and S.Shenker, "Analysis and simulations of a fair queuing algorithm," in *Proceedings on Communications Architectures and Protocols*, Austin, TX, USA, pp. 1-12, September 1989.

[2] A. Parekh and R. Galleger, "A generalized processor sharing approach to flow control in integrated services networks: The multiple node case," in *Proceedings on IEEE INFOCOMM*, pp. 521-530, March 1993.

[3] M.Shreedhar and G.Varghese, "Efficient fair queuing using deficit round robin," in *Proceedings of ACM SIGCOMM*, pp. 231-242, August 1995.

[4] L. Zhang, "Virtual clock: A new traffic counter algorithm for packet switched networks," in *Proceedings of ACM Transactions on Computer Systems*, pp. 101-125, May 1991.

[5] B. Suter, T.V. Lakshman, D.Stiliadis, and A.Choudhury, "Design considerations for supporting tcp with per-flow queueing," in *Proceedings of INFOCOMM*, pp. 299-306, April 1998.

[6] K. Nichols, V. Jacobson, and L. Zhang, "A two-bit differentiated services architecture for the internet," in http://diffserv.lcs.mit.edu/Drafts/draft-nichols-diff-svc-arch-00.txt, Internet Draft, December 1997.

[7] D.D. Clark and W. Fang, "Explicit allocation of best-effort packet delivery services," http://diffserv.lcs.mit.edu/Papers/exp-alloc-ddc-wf.pdf, November 1997.

[8] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," in *IEEE/ACM Transactions on Networking*, pp. 397-413, August 1993.

[9] J. Heinanen, T. Finland, and R. Guerin, "A three color marker," ftp://ftp.ietf.org/rfc/rfc2697.txt, Internet Draft, February 1999.

[10] J. Heinanen, T. Finland, and R. Guerin, "A two rate three color marker," ftp://ftp.ietf.org/rfc/rfc2698.txt, Internet Draft, March 1999.

[11] I. Yeom and A.L.N. Reddy, "Realizing throughput guarantees in a differentiated services network," in *IEEE Int. Conf. on Multimedia Computing and Systems*, pp. 372-376, June 1999.

[12] J. Ibanez and K. Nichols, "Preliminary simulation evaluation of an assured service," http://iamwww.unibe.ch/~balmer/papers/draft-ibanez-diffserv-assured-eval-00.txt.gz, Internet Draft, August 1998.

[13] A. Basu and Z. Wang, "A comparative study of schemes for differentiated services," in *Bell Labs Technical Report*, February 1998.

[14] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the internet," in *IEEE/ACM Transactions on Networking*, pp. 458-472, August 1999.

[15] I. Yeom and A.L.N. Reddy, "Modeling tcp behaviour in a differentiated services network," Technical Report 23, Department of Electrical and Computer Engineering, Texas A&M University, May 1999.

[16] S. Sahu, P. Nain, D. Towsley, C. Diot, and V. Firoiu, "On achievable service differentiation with token bucket marking for tcp," in *Tech. Report 99-72, University of Massachusetts*, December 1999.

[17] I. Stoica, S. Shenker, and H. Zhang, "Core-stateless fair queueing: Achieving approximately fair bandwidth allocations in high speed networks," in *Proceedings*

of ACM SIGCOMM, pp. 118-130, October 1998.

[18] D. Tong and A.L.N. Reddy, "Qos enhancement with partial state," in *International Workshop on QoS*, UCL, London, UK, pp. 87-96, June 1999.

[19] L. Peterson and B.Davie, *Computer Networks: A Systems Approach*, vol. 1, Morgan-Kaufman Publishers, San Francisco, CA 94104, 1999.

[20] K.C. Claffy, G.C. Polyzos, and H-W. Braun, "Application of sampling methodologies to network traffic characterization," in *Proceedings of ACM SIGCOMM, pp194-203*, May 1993.

[21] T.J. Ott, T.V.Lakshman, and L.H. Wong, "Sred: Stabilized red," in *Proceedings of IEEE INFOCOM*, pp. 1346 -1355, March 1999.

[22] W. Feng, D.D. Kandlur, D. Saha, and K.G. Shin, "Adaptive packet marking for providing differentiated services in the internet," in *Proceedings of Int. Conf. on NetworkProtocols*, pp. 108 -117, October 1998.

[23] M. Goyal, A. Durresi, P. Misra, C. Liu, and R. Jain, "Effect of number of drop precedences in assured forwarding," in *Tech. Report, Ohio State University*, March 1999.

[24] M. Parris, K. Jeffay, and F.D. Smith, "Lightweight active router-queue management for multimedia networking," in *Proceedings of SPIE Conf. on Multimedia Computing and Networking*, pp. 162-174, January 1999.

[25] D. Lin and R. Morris, "Dynamics of random early detection," in *Proceedings of ACM SIGCOMM*, pp. 127-137, October 1997.

VITA

Saikrishnan Gopalakrishnan was born on July 10, 1975 in Namakkal, India. He received his Bachelor of Engineering degree in Computer Science and Engineering from Government College of Technology, Coimbatore, India in May 1996. He worked for 2 years in Wipro Systems, Bangalore, India and joined the master's program in Computer Science at Texas A&M University in August 1998. At Texas A&M his research has been focussed on issues related to problems in differentiated services networks. His address is Department of Computer Science, Texas *A&M* University, College Station, Texas 77843.

The typist for this thesis was Saikrishnan G.