Multipath Aware TCP (MATCP)

Peter Dimopoulos (dimpet@cs.rmit.edu.au)
Panlop Zeephongsekul (panlopz@rmit.edu.au)
Zahir Tari (zahirt@cs.rmit.edu.au)
RMIT University
P.O. Box 2476V Australia 3001

Abstract

On the Internet many different paths exist between each source and destination. When single path routing is used these paths can be under utilized, not used fairly or not used at all. One way to overcome this is to allow multipath routing. But when multiple paths are used TCP congestion control can be negatively affected and cause poor goodput performance due to the reordering of packets. We propose MATCP (Multipath Aware TCP) which makes modifications to TCP that allows it to monitor and select which path it takes through the network for each flow. MATCP is compared to single path routing and is validated using extensive simulation. MATCP is found to greatly improve fairness between flows while providing equal or better utilization of links than single best path networks. [1]

1 Introduction

Multiple paths occur often on the Internet and in service provider networks. For a service provider, having a single high-speed link can be a reliability risk and many service providers require their networks to have multiple redundant links to the external Internet and internally.

Some Internet routing protocols only configure a single best path, even though many alternate paths may exist. In fact in Savage et al [7] it is shown that in 30 to 80 per cent of cases a better alternate path exists. Some routing algorithms allow the best path to change as network conditions change, which in effect produces a type of dynamic routing. Newer routing algorithms allow more than one path to be used at the same time, for example, when there are two paths that have equal cost. Using multiple paths allows the network to be utilized more efficiently because network paths can be shared more easily. For example there may be two paths, one being used at 100% utilization, the other at 20%. With multipath routing each path could split the load at 60% each. Even though this may not seem an advantage, when considering TCP connections, it will be, because a TCP connection will attempt to use as much of the bandwidth as is available. The TCP connections on the 100% utilized link would not find any more bandwidth, but with multipath the free bandwidth on the underutilized link can be accessed.

In current multipath networks the TCP layer is not aware of the route which its packets take. The routing decisions are made at each hop (router) at the IP layer and routes are decided by routing algorithms. These routing decision can be based on a number of criteria like hop count, bandwidth available or shortest delay. Because the routing decisions are made at any router of the network at any time, it is possible for packets of a TCP connection to take different paths. When packets take different paths, packet reordering can easily occur, for example, when one path has higher delay than another the first packet could enter the long link and the second the short link; the second packet would then arrive at the destination before the first packet.

Packet re-ordering is harmful to TCP throughput [3, 2, 6] for many reasons including causing duplicate acknowledgments when packets arrive out of order. Duplicate acknowledgments then cause retransmission of packets that are not lost in the first place, just out of order. It is possible to make a routing decision at each router which is TCP flow specific: this allows packets of a flow to all take the same path. To do this, each router would need to keep information about every single flow that traverses it. When there is a large number of flows this can be expensive in terms of processing and memory.

Given the difficulty of keeping TCP packets in order within flows, it would be an advantage for each TCP source to be able to choose which route each of its packets takes. In this way it could guarantee order within a flow by choosing the same route for all its packets. Of course the entire end-to-end path of every source to every destination on the

[1] This project is fully supported by the ARC Discovery Grant no. DP0346545 awarded by the Australian Research Council (ARC) for 2003-2005 and Sun Microsystems.

Internet can not be pre-determined, but it is possible to predetermine routes through each smaller service provider network. It is most important to pre-determine critical routes, like those links that traverse the service provider boundaries.

An example of this type of system is shown in Figure 1b where two disjoint paths exist through a service provider network. The end point of the links are not in the service provider network, but the bottleneck is assumed to be part of the service provider network. The internal network is assumed to have more than adequate bandwidth so as to minimize any internal bottlenecks. This would be the case in many service provider networks where it is not so expensive to have high bandwidth links between equipment that is closely located. The problem then becomes how to optimize the utilization of multiple high cost links (for example, links to the external Internet) and provide fair goodput to TCP sources.

Multipath TCP (MATCP) allows the path used by a flow to be selected at the TCP layer. Each flow is labeled with a path number from one to M where M is the maximum number of paths. The path selection is done using a selection algorithm which makes use of information which is constantly collected by TCP, like RTT (Round Trip Time) and packet drops. This work concentrates on the case of a single service provider with multiple outgoing links; each of these is labeled with one of the M path numbers. The service provider could signal the maximum number of paths back to the TCP source or a standard/fixed number of paths could be used. Allowing the selection of path at the TCP layer allows each flow to take a different path rather than all the flows of each source taking the same path. This means a finer grain of load-balancing can be achieved. Making the path selection at the sources also greatly reduces the the complexity and state required in the network. For example, routers do not need to keep any state information about flows or sources to make routing decisions.

The following is a summary of the contributions of this chapter

- We introduce the Multipath TCP protocol which allows TCP to select which path its flows take from a number of pre-numbered routes. This pushes network complexity out to the source reducing the need for expensive (in terms of memory and processing) hardware.

- We provide a more fine grained control of load balancing by allowing each TCP connection to take a different route. In current literature only load balancing on source/destination IP pairs is viable.

- We implement and verify the fairness and goodput performance of the MATCP protocol using simulation.

In Section 2 the architecture and design of the proposed protocol is presented. This is followed by simulation results in Section 3, Future work in 4 and a conclusion in Section 5.

2 Architecture

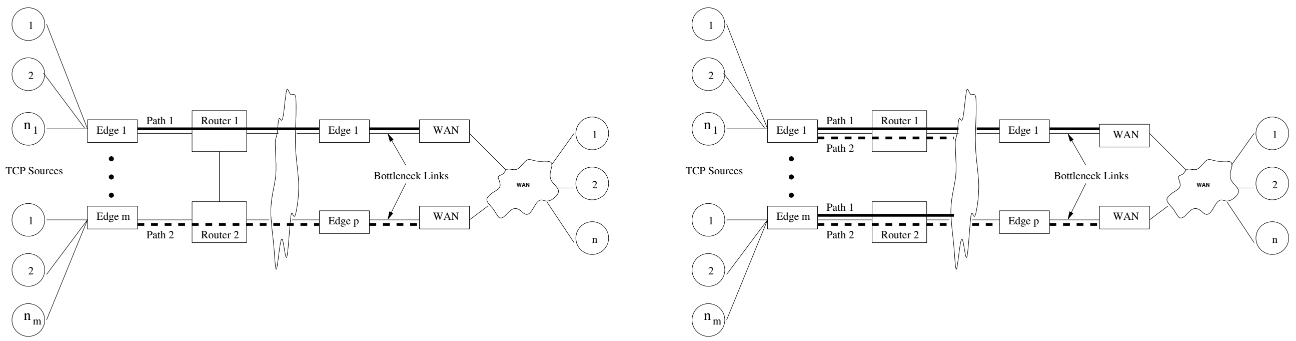
2.1 Single Path TCP

Figure 1a shows a typical single path system where a number of TCP sources are tied to an edge router upon connection to the network. Each of the m edges has n_m active customers connected to it at a time. The number of active customers n_m is assumed to be constant over short time periods (tens of minutes) and have a Uniform distribution. The p bottleneck links are at the edge of the service provider network and connector to the wider network. Each of the m edges and all its customers are allocated to one of p links. Allocating each edge or even customer statically to a specific link will cause poor utilization of links because some customers will require more bandwidth than others. For example if edge one has ten customers and they are all using path one, and edge two has three customers and they are using path two, then the edge two customers will be able to obtain higher bandwidth than edge one customers because only three of them are sharing the entire path two bandwidth.

2.2 Multipath Aware TCP (MATCP)

The multipath system shown in Figure 1b provides better utilization of both links and better fairness to sources. The extra paths allow each source to choose which path to use. This decision could be made in two places; at the edge router [1, 5] or at the customer end-point. Tracking every sources flows at the edge would be required if a path decision is to be made at the edge. Rather than do this, the choice of path is made at the TCP source. The TCP source could learn the p paths when the logical connection is established to the edge or whenever a new path is created or removed.

The problem now becomes how to select which path each TCP flow should take. To make a decision on which path is best for each flow it is important to be able to compare the quality of all the paths. TCP currently uses acknowledgment packets to estimate the congestion in a link. Congestion is defined by packet loss and round trip delay. If there are multiple known paths then it is possible to use these same estimates to determine the congestion of each of the paths. Obviously these estimates are not available if only one path is in use. Splitting a single TCP connection over multiple paths has also been shown in current research



(a) Singlepath

(b) Multipath

Figure 1. Example of a Single and Multipath aware TCP network

to reduce throughput due to packet re-ordering [6, 3]. This means each flow should follow only a single path.

Three alternative ways to deal with this problem are

1. In systems where each end-point has many short TCP flows, route each flow weighted round-robin-style through alternate paths while allocating a higher weight to the better path. All the paths must be used so that some measurement can be made of the paths quality.
2. Use a different path for each round trip of a flow. This reduces the amount of packet reordering while allowing more measurements of each of the paths.
3. Share information between end points about the different paths. This would involve some communications overhead between local nodes that share the same paths.

This work focuses on the first of these alternatives, where a TCP flow is designated to a particular path based on some path selection criteria.

2.2.1 Path Selection criteria

Path selection is concerned with selecting the most suitable path for each TCP flow in order to improve the utilization of the network, fairness between TCP sources, and goodput of each TCP source. Many different measures can be used in path selection including a history of RTT, RTT variation and number of packet drops.

RTT variation selection

A simple way to select a path is using the RTT variation which is already estimated in every TCP session. The RTT is made up of two main components: propagation delay and RTT variation (queuing delay). The queuing delay changes as congestion in the network changes; higher delay occurs when there is more congestion. In a multiple bottleneck network, false reading of a particular path may also occur because the external bottleneck is causing the variation rather than the internal path bottleneck. This work concentrates on the case where the internal path bottleneck causes a much greater delay and loss than the external bottleneck. This is of course a limitation of this algorithm which should be a subject of future work.

The simplest algorithm would be to always choose the path with the smallest RTT variation. But for TCP to collect information about the other paths, they must be used as well. To make sure that all paths are used, a refresh constant R is defined. After a path has been used R times in a row, every paths RTT is set to zero which forces each path to be used. A path with zero RTT will be lower than any of the other paths as they are measured. The refresh constant then controls how up-to-date information is: a high R will lead to less up-to-date information that a low R. If R is too low then some paths that are not good may be used too often because they are constantly being checked. The number of paths will also affect how R is set, because the more paths there are, the more time is spent using each path to get measurements.

Packet drop selection

The number of packets that have been dropped on a path can be used to select the best path. A path with many drops should be selected less than a path with few drops. This works in a similar way to the RTT variation as explained above.

RTT Selection

The RTT could also be used in selecting the path: this allows shorter propagation delay paths to be selected before longer paths. The selection then does not just rely on the load of the path. RTT selection is most useful when the RTT of different paths is not the same which is the case in

0-7695-2588-1/06 $20.00 © 2006 IEEE

Table 1. Example of Probability Assignment

Path	RTT	Probability
1	100	$\frac{326-100}{226+211+215} = \frac{226}{652}$
2	115	$\frac{326-115}{226+211+215} = \frac{211}{652}$
3	111	$\frac{326-111}{226+211+215} = \frac{215}{652}$

many paths on the Internet. Some paths may have a larger number of hops or longer links which cause longer propagation delays. The RTT variation selection method exploits changes in queuing delay, but RTT selection exploits changes in queuing delay as well as different propagation delays. For example, suppose two paths exist, one with a short propagation delay and the other with a long propagation delay. The short propagation delay path should be used as long as its propagation delay plus queuing delay is smaller than the longer path's propagation delay plus queuing delay. If the shorter path's queuing delay is large then it is possible that the longer propagation delay path actually has a smaller total delay. The RTT selection method may not work properly when the queuing delay never increases the total RTT delay above the propagation delay of the longer path. This would occur when a buffer is not large enough and would lead to the shorter RTT path always being selected even though it is congested. A possible solution to this problem is to use weighted RTT selection.

Weighted RTT Selection

Rather than just select the lowest RTT path and a refresh constant to probe paths we can assign a probability of selecting each path. The probability would have to be related to the RTT measurement. One possible way to create the probability is to use Equation 2 whose use is illustrated in Table 1. In this way the longest RTT gets the smaller probability and the shortest RTT gets the bigger probability of seleciton. With this method there is no need for a refresh constant because each path will be used with a certain probability. Obviously a problem could occur if a path has a very small probability of selection because it will be used much less often and therefore measurements for that path will be not be as up-to-date.

$$RTT_{total} = \sum_{y=0}^{y=n} RTT_y \quad (1)$$

$$W_x = \frac{RTT_{total} - RTT_x}{\sum_{y=0}^{y=n} RTT_{total} - RTT_y} \quad (2)$$

2.3 Interoperability and Implementation

MATCP can easily be introduced into single path network. For example if there is a number of single path TCP sources which have a preset path and the MATCP sources run concurrently with them, then MATCP will select paths with the lightest load which would most likely be the ones the single path TCP sources are not using. The more MATCP sources the more evenly the load will be distributed over the multiple paths. This means that MATCP can gradually be introduced into the network. A user option to enable MATCP could easily be added which allows the user to take advantage of multiple paths if they exist. Sources that don't implement MATCP will just work as normal.

MATCP would require the edge router to communicate which paths are available to the end user. This could be done on the initial connection setup, possibly when the link is brought up using PPP. For example in an MPLS network the edge router could pass a label for each path that is available to the end user. This would be like extending all the existing paths to the end user where the choice of path can then be made of which path to use. If the paths changes, update messages could be sent to inform the end point of new paths, or removed paths. A new but very simple protocol would be required to provide this functionality. IPv6 also provides some functionality that could allow specific paths to be selected but this has not been thoroughly examined but could be future work.

2.4 Reliability of paths using retransmission timeouts

The TCP retransmission timer can be used to assess the reliability of the multiple paths. The number of timeouts that have occurred on a path indicate high congestion or a complete disconnection of the path. Because MATCP is based on TCP performance, MATCP automatically learn which links are disconnected because these links perform very badly. Links that perform very badly will rarely be selected by MATCP. In this way MATCP will automatically solve the problem of unreliable links by not using them.

3 Simulation Results

The four different algorithms presented in Section 2.2.1 are simulated using the NS-2 Simulator and compared to a single path TCP network. In the single path network each TCP source has a pre-assigned outgoing edge router and must always use that router when accessing the Internet. The topology used in the simulations is shown in Figure 1b, while Table 2 shows the default parameters used in all simulation unless otherwise specified. All simulations are run for 500 seconds which we consider long enough to reach a steady state. The goodput results are normalized to the capacity of the links. For example if the link capacity is 10Mbps then a 1Mbps result will be given the value 0.1. We now investigate the outgoing link utilization, average TCP source goodput and variation of TCP source goodput. The goodput variation of each source gives an idea of the fairness of bandwidth distribution over the outgoing links. A low variation means that all sources are receiving a fairer share of the bandwidth.

Table 2. Default Values for Simulations

Description	Variable	Value
TCP Protocol		Sack
Window Size	W	16
Average Burst Time	B	8 seconds
Average Idle Time	I	6 seconds
Buffer Size	K	100 Packets
Buffer Maxp		0.1
Buffer Min_{th}		25
Buffer Max_{th}		50
Packet Size		1460
Bottleneck Capacity	C_x	40Mbps
Propagation Delay	T	65ms
Avg Num Connections	N_x	80
Var Num Connections	V	40
Number of Paths	P	4

3.1 Uniform distribution of sources per edge

3.1.1 Average Number of TCP sources

Figure 2 shows the effect of changing the average number of TCP sources on each edge. The number of sources on each edge is selected from the uniform distribution with the indicted average and a fixed variation. The variation is defined by the range in which the number can be selected from. For example, when we say a variation of 20 with an average of 80 this means that the number of sources is uniformly distributed between 60 and 100. A variation of 40 would be a uniform distribution of between 40 and 120 sources. In this figure the variation is 40.

The first row of Figure 2 shows the goodput of each path with the solid lines denoting a single path system and the dotted lines denoting each path for the MATCP algorithms. Each column presents one of the path selection algorithms (Drop, RTT Variation and Weighted RTT variation). Since RTT is the same in these experiments RTT variation will give the same results. The goodput produced on each path using the single path method is seen to vary greatly between each path, especially when the average number of sources is small. Whereas the goodput for all paths of the MATCP algorithms is very similar in all cases. Even though they are very similar, the different MATCP algorithms can be seen to provide slightly different goodputs.

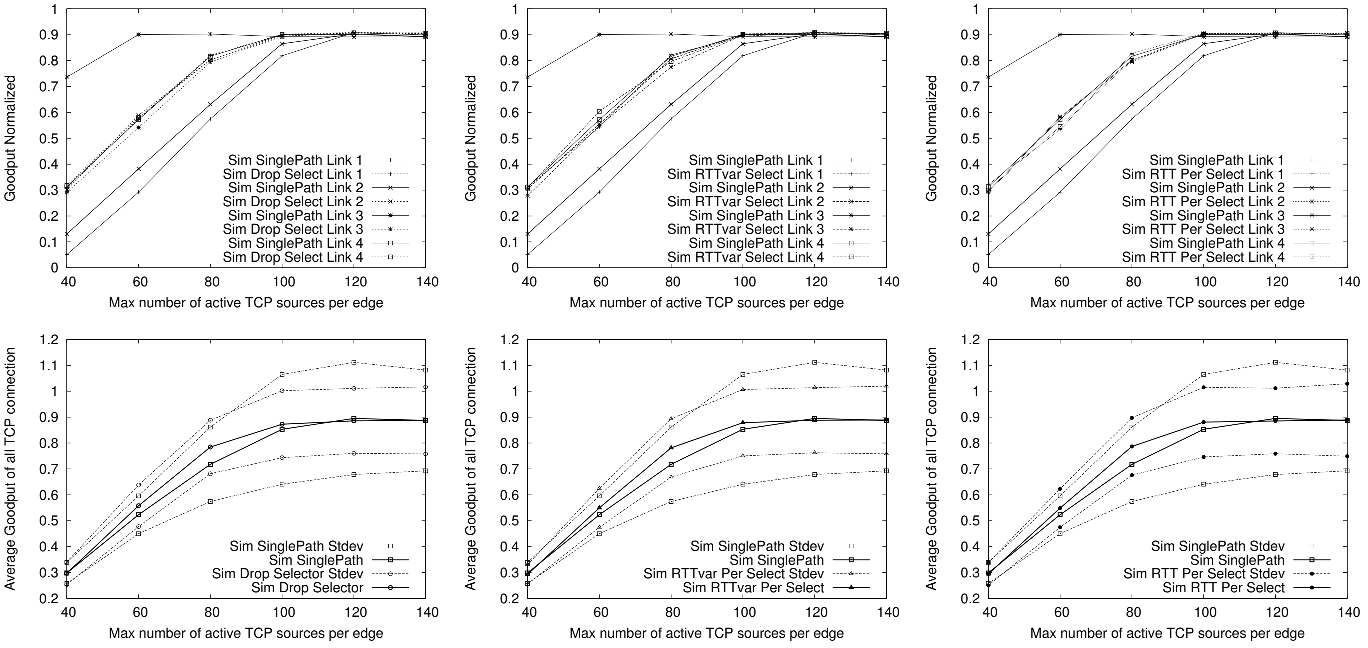
The second row of Figure 2 shows the average goodput of all TCP sources and one standard deviation from the average on either side. We use the standard deviation as a measure of fairness between TCP sources. A smaller standard deviations means more of the sources receive roughly the same bandwidth. The standard deviation for the MATCP algorithms is almost half that of using a single path when there is a large number of sources. This means that MATCP is providing better fairness to the TCP sources at high loads. The conclusion here is that MATCP can slightly improve goodput at low loads and improves fairness at high loads.

3.1.2 Variation of Number of TCP sources

Figure 3 shows how MATCP performs much better than a single path when the variation of the number of TCP sources between the edges is large. The first row of the figure shows that MATCP actually shares the links better or equal in terms of goodput at all variations. As the variation increases, the performance of MATCP improves compared to using a single path. This is because when the variation is high, some routers end up having a large number of sources whereas some have a very low number of sources active. In the second row we see that the average goodput of the MATCP algorithms improves on the single path as the variation increases. The standard deviation of the MATCP goodput is also almost half that of the single path.

3.1.3 Different Idle and Burst Time

The frequency of file transfers (flows) can affect the performance of the MATCP algorithm because it determines how often each of the separate paths are probed. If the paths are not probed frequently then the decision of which path to select will be made from old information. The average idle time determines the time between flows and therefore the frequency of flows. In Figure 4 on the second row we see that the standard deviation of source goodputs increases as the idle time increases. This shows that MATCP does not perform as well in terms of fairness when the frequency of connections is lower. The difference in goodput of each path is still closer than using a single path as shown in the first row. It is also found that larger the burst sizes provide better the fairness; this is because larger burst sizes allow the RTT variation, drop or RTT to be measured more accurately and a better path to be selected. Further analysis of burst size can be found in [4].



(a) Drop **(b) RTT Variation** **(c) WRTT**

Figure 2. Uniform Distribution: Different number of TCP sources

3.2 Different RTT on each Path

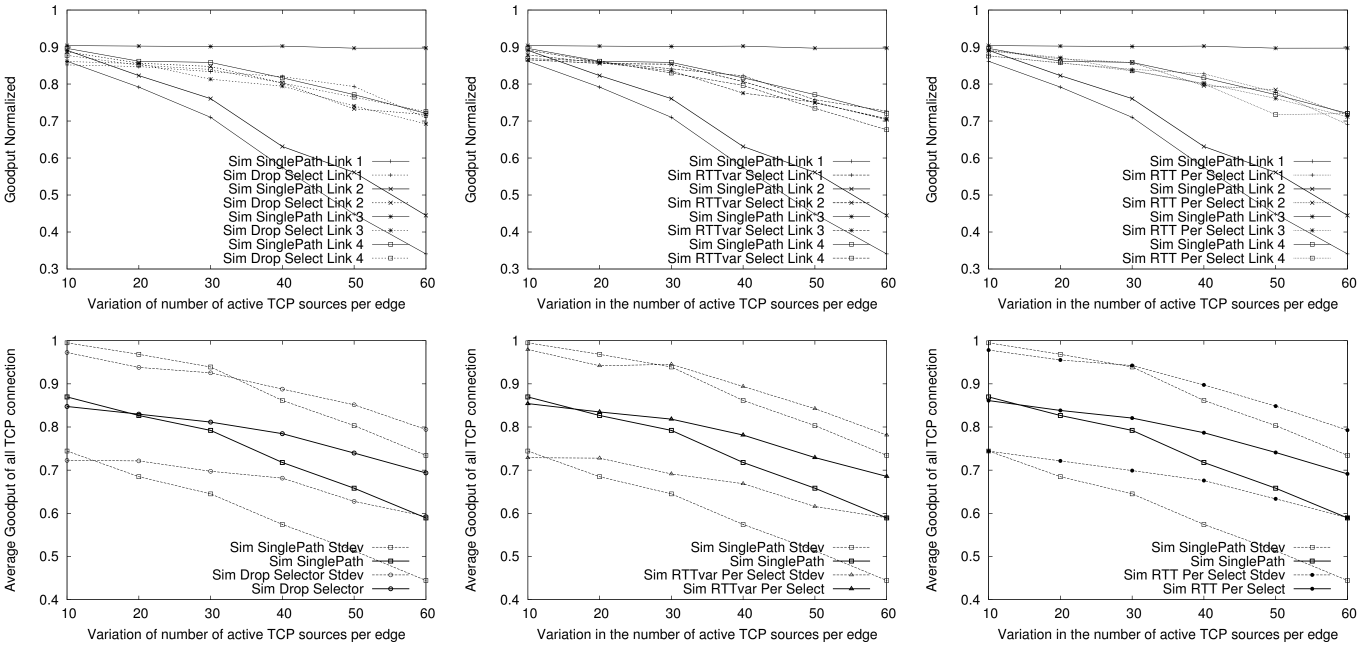
Here a different propagation delay is assigned randomly to each bottleneck link. The propagation delay value is drawn from a uniform distribution so that the variation in delay can be changed without changing the average delay. Propagation delay is important in TCP because a short delay allows the congestion window to increase faster and therefore obtain a higher goodput in a shorter time.

3.2.1 Average Number of TCP sources

The RTT goodput plots in Figure 5b shows that longest RTT (52ms) has the lowest utilization at all loads. This is because this link rarely has the shortest RTT even though the other links have a higher utilization and therefore queuing delay. RTT selection is therefore not a good selection method when the propagation delays between the links vary. Both RTT variation (Figure 5a) and WRTT (Figure 5c) perform much better than RTT in keeping the goodputs of all the links at a similar level. The WRTT performs the best because it considers both the propagation delay and queuing delay. Then it evenly distributes the load across all the links based on their weights. Notice that each of the goodputs are in order of there propagation delay, with the highest propagation delay getting the least goodput. The bottom row of Figure 5 shows that all the MATCP selectors provide a lower deviation from the mean than the single path method. This again means that multipath is providing better fairness to each TCP source. The WRTT actually has a slightly higher average goodput at lower loads which shows it is utilizing the links better than the single path.

4 Future Work

In future work we will look at extending the network beyond just the access architecture of Figure 1. For example each packet can be marked with a number 1 to M. The internet will then consists of a number of interconnected networks. Each of these networks can provision N paths through their own network for each network that surrounds it. When a packet arrives at a network boundary its marked number is used to select one of the N paths. Label swapping may be required at the network edges to allow the best paths to be used. If less than M paths exist through the network or the network does not support label swapping then the system falls back to the best existing paths. This allows an end-to-end network of multiple paths to exist with TCP controling and monitoring which paths are used. Reliability will also be investigated through the use of the TCP timeout mechanism. For example if a path times out many times then a new path can be selected.



(a) Drop **(b) RTT Variation** **(c) WRTT**

Figure 3. Uniform Distribution: Variations in the number of TCP sources per Edge

5 Conclusion

MATCP addresses issues concerned with using TCP over networks where multiple paths exist. Problems like packet re-ordering where packets take a different routes and arrive out of order are addressed and overcome by allowing TCP to make the decision of which path an entire flow of packets will take. Making decisions at the source MATCP allows routers to have less state and complexity while achieving the same level of load balancing granularity. The utilization and fairness of the network is also analysed and MATCP is found to provide fairer sharing of network resources than a single best path network.

References

[1] O. Alparslan, N. Akar, and E. Karasan. AIMD-based online MPLS traffic engineering for TCP flows via distributed multi-path routing. *To Appear in Annales Des Telecommunications*, 2004.

[2] J. Bennett, C. Partridge, and N. Shectman. Packet reordering is not pathological network behavior. *IEEE/ACM Transactions on Networking*, 7(6):789 – 798, December 1999.

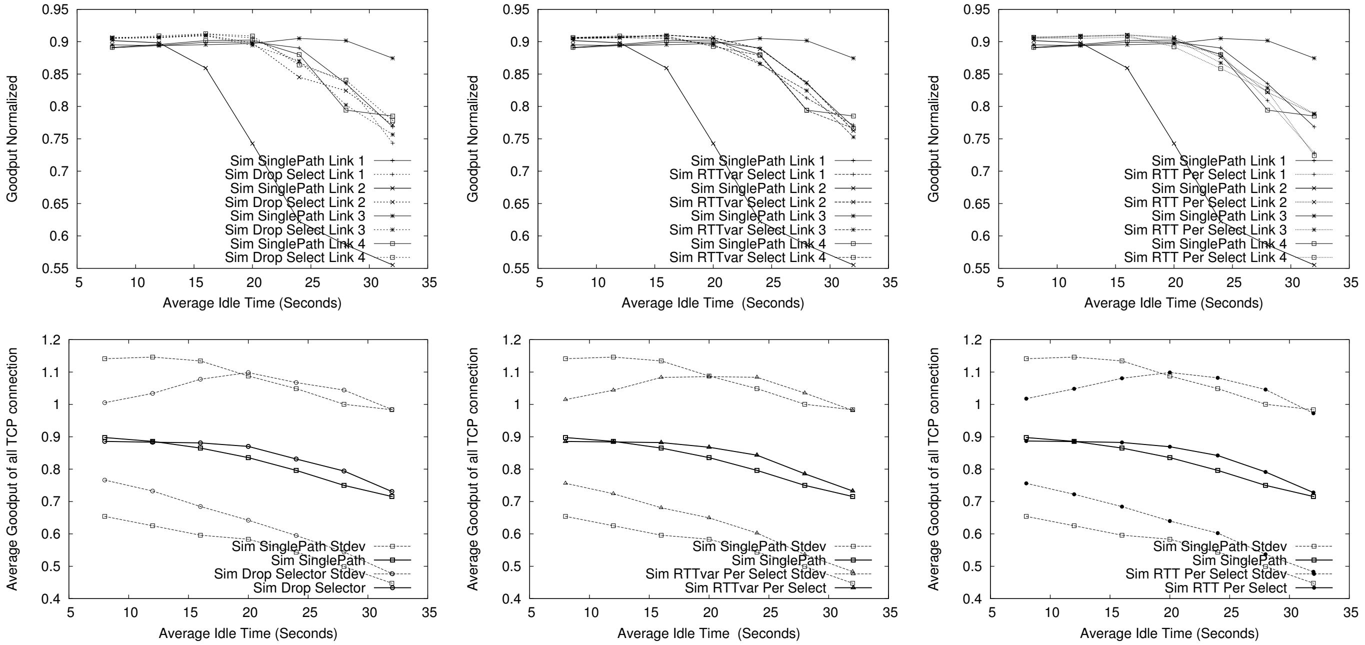
[3] E. Blanton and M. Allman. On making TCP more robust to packet re-ordering. *SIGCOMM Computer Communication Review*, 32(1):20–30, 2002.

[4] P. Dimopoulos, P. Zeephongsekul, and Z. Tari. Multipath aware TCP (MATCP). Technical Report TR-06-1, RMIT, March 2006.

[5] A. Elwalid, C. Jin, S. Low, and I. Widjaja. MATE: MPLS adaptive traffic engineering. In *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 3, pages 1300–1309, April 2001.

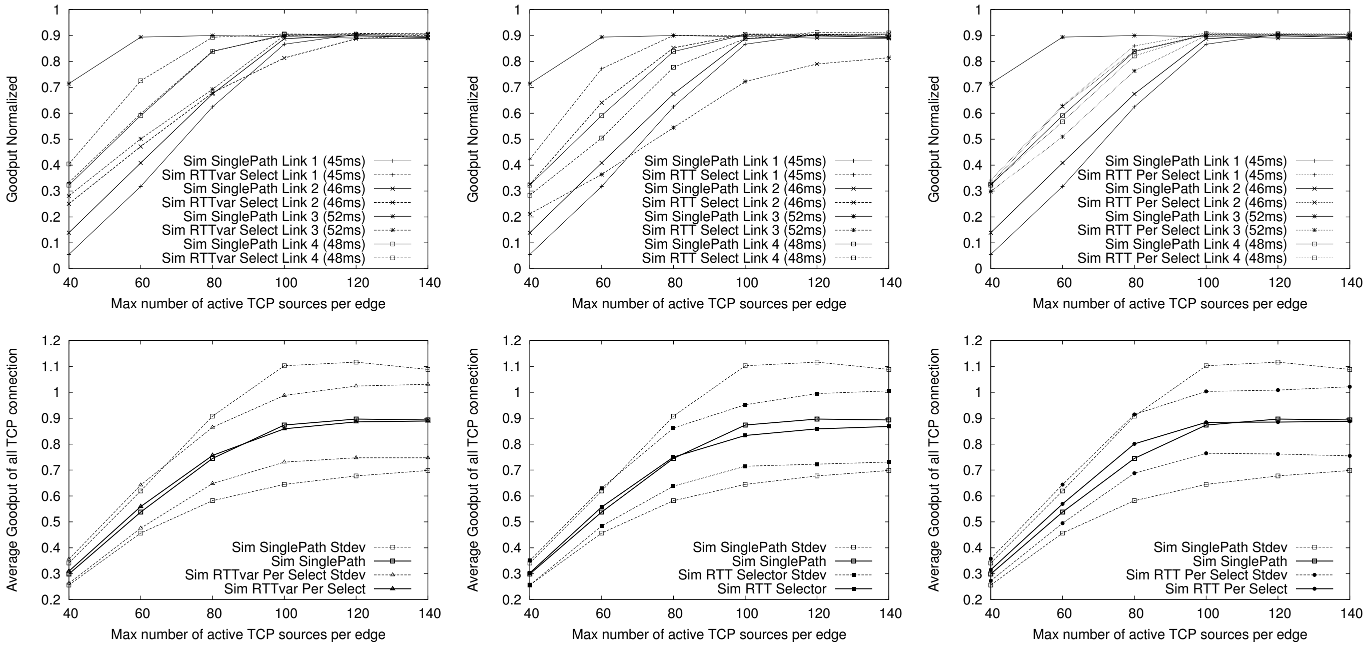
[6] C. Ma and K. Leung. Improving TCP reordering robustness in multipath networks. In *Proceedings of the 29th Annual IEEE Conference on Local Computer Networks (LCN)*, pages 409–410, November 2004.

[7] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson. The end-to-end effects of internet path selection. *SIGCOMM Computer Communication Review*, 29(4):289 – 299, October 1999.



(a) Drop **(b) RTT Variation** **(c) WRTT**

Figure 4. Uniform Distribution: Different Idle times



(a) RTT Variation **(b) RTT** **(c) WRTT**

Figure 5. Multi RTT: Different Number of TCP sources