Impact of Delay Variability on LEDBAT Performance

Amuda James Abu and Steven Gordon
Sirindhorn International Institute of Technology, Thammasat University,
Pathumthani 12000, Thailand.
james@ict.siit.tu.ac.th, steve@siit.tu.ac.th

***Abstract*—Low Extra Delay Background Transport (LEDBAT) congestion control algorithm is designed to address the unfairness problem of TCP aggravated by applications that use multiple TCP connections for data transfer. LEDBAT operates under the assumption that the queue delay at the access router of the bottleneck link will be the primary varying contributor to end-to-end one-way delay. However this assumption will not hold if a route change occurs, which causes significant variations in the path delay. This paper analyses the impact of route changes on LEDBAT throughput and fairness. In addition to a formal description of the behaviour of LEDBAT congestion window when route changes, we present results from simulations showing the negative impact of route changes on throughput for a LEDBAT source and fairness with other sources. Importantly, our analysis shows that more work is needed to improve LEDBAT performance in the case of route changes before the novel algorithm can be considered as a suitable congestion control algorithm in the Internet.**

***Keywords*-LEDBAT, delay variability, congestion control, peer-to-peer file sharing, route changes.**

I. Introduction

Motivated by the unfairness problem in TCP aggravated by applications that use multiple TCP connections to transfer data in the Internet, the Low Extra Delay Background Transport (LEDBAT) congestion control algorithm has been developed as an alternative for such applications [1]. By using multiple TCP connections an application from one customer can induce significant queue delays in the ISP's access router, severely impairing the performance of voice, video and gaming applications of other customers. Therefore a LEDBAT source adjusts its sending rate to maintain the queue delay in the access router at or below a pre-defined target. The LEDBAT source aims to saturate the bottleneck link in the path to the destination in order to achieve high throughput for applications when no other traffic exists. A competing aim is also to yield quickly when other (TCP or UDP) sources start sending over the bottleneck link. Micro Transport Protocol (uTP) [2] is an application layer congestion control protocol used by μTorrent (a widely used UDP-based BitTorrent protocol) and similar to LEDBAT [1].

LEDBAT is a distributed one-way delay-based algorithm, designed to provide a lower-than-best-effort service for background file transfer applications, especially P2P file sharing. It operates under the assumptions that the queue delay at the access router of the bottleneck link will be the primary varying contributor to end-to-end one-way delay, and the access router does not employ Active Queue Management (AQM). This is typical in numerous ISP networks [3], [1]. To measure the queue delay a LEDBAT source adds a timestamp to each data packet sent; the receiver calculates the one-way delay, returning the value in acknowledgement packets. The source then estimates the queue delay from the difference between the current one-way delay and a base one-way delay calculated during a given period of time rather than the start of the connection. The LEDBAT source controls its sending rate by updating its congestion window[1], w, for every ACK received:

$$w(t+1) = \begin{cases} \frac{1}{2}w(t) & \text{if packet loss} \\ w(t) + G\frac{d_{tar} - \hat{d}_{que}(t)}{w(t)} & \text{otherwise} \end{cases} \quad (1)$$

where $\hat{d}_{que}$ is the current estimated queue delay, d_{tar} is the target delay and G is the gain. d_{tar} and G (both constants) are two key parameters that influence how well LEDBAT achieves its aims of saturating the bottleneck and yielding quickly to other traffic [5], [6].

The LEDBAT congestion control algorithm in [1] assumes that only queue delay varies while other components of delay are approximately constant. This is not always true in a typical Internet environment [7]. However, route changes in the Internet can in fact cause the time it takes to send a packet from the source to destination to vary, excluding the time spent in the queue. This can be due to different link delays and consequently path delays existing on different routes in the Internet. By link delay, we mean the time it takes to get an IP packet across a link. This includes transmission, waiting, and retransmission by the data link layer protocol. The path delay (d_{path}) is simply the sum of the link delays in the access (d_{access}) and core (d_{core}) networks and queuing/processing delay at routers (d_{que}). When using the same route, small variations in the path delay are possible because of varying link and queuing delays. But when a route changes, large changes in the path delay may occur. Therefore, route changes play a significant role in delay variability. We assume only the change in path

[1]As with TCP, the LEDBAT sending rate and congestion window are approximately proportional [4]

delay due to route changes significantly contributes to delay variability beyond an ISP network [7].

As far as we know, no work has analysed the impact of route changes on the throughput and fairness of LEDBAT. This is opposite to the works in [8], [9] that showed the impact of route changes on the performance of TCP-Vegas, a delay-based congestion control algorithm similar to LEDBAT. Therefore, our work is significant towards considering LEDBAT as a suitable congestion control algorithm in the Internet.

In this paper, we analyse the impact of route changes on LEDBAT throughput and fairness under different network conditions. We start by mathematically describing the behaviour of LEDBAT congestion window when route changes, used to support the discussion of our analysis results. Our results from simulations show that LEDBAT throughput is negatively affected when route changes result in an increasing one-way path delay while the LEDBAT fairness objective of keeping queue delay as low as the target is compromised for the case of decreasing one-way path delay. Additional results indicate that decreasing the average time between successive changes of path delay causes LEDBAT throughput to decline with increasing average magnitude of change of path delay. Importantly, our analysis show the need for more work to improve the performance of LEDBAT in the case of when route changes.

The rest of this paper is organized as follows. Section II reviews recent works on LEDBAT. The description and assumptions of our network scenario and a model of LEDBAT congestion window when route changes are given in Sections III and IV, respectively. LEDBAT performance is analysed via simulations in Section V with results showing the negative impact of route changes on the performance of LEDBAT. Concluding remarks are given in Section VI.

II. Related Work

The problem of congestion in a network started back in the 80s [10]. Ever since then, numerous congestion control algorithms have been proposed in the literature and some of which are used in the Internet today. They can be grouped as: loss-based, delay-based, rate-based, and low-priority. LEDBAT exhibits characteristics of delay-based and low-priority algorithms. However, the novel algorithm differs from many such algorithms including TCP-Vegas [11], TCP-NICE [12] and TCP-LP [13], in that it aims at minimizing queue delay in a network to a defined value that can be tolerated by voice, video and gaming applications. A survey of these and other low-priority congestion control algorithms has been provided in [14]. Although LEDBAT is relatively new, research and experimentation of its operation have begun. A review of the recent works on LEDBAT is given as follows.

The works in [15], [16], [17], [5], [18], [6] show that LEDBAT achieves some of its design objectives, but not without some challenges under certain conditions. In [15], the authors evaluated LEDBAT performance in a controlled testbed and Internet experiment and found that TCP traffic on the ”unrelated” backward path is capable of causing LEDBAT to significantly underutilize the link capacity in the forward path. It has been shown in [16] that LEDBAT competes fairly with TCP in the worst case (i.e. LEDBAT misconfiguration). Potential fairness issues such as latecomer advantage between LEDBAT flows have been identified in LEDBAT [16] which can be fixed by using slow-start in the LEDBAT algorithm. In addition to the proposed solution for the LEDBAT intra-protocol unfairness by [16], the authors of [17] proposed that random drops of LEDBAT sender window and multiplicative decrease are promising solutions to the problem of LEDBAT latecomer advantage. The proposed solutions are not without a performance trade-off between link utilization and fairness. Comparative analysis of LEDBAT with other low priority protocols (TCP-NICE and TCP-LP) in the presence of TCP showed that LEDBAT achieves the lowest priority [5]. The authors further showed via sensitivity analysis that unfairness exists between two LEDBAT flows with different delay targets or different network conditions and LEDBAT is aggressive with TCP in the case of LEDBAT misconfiguration. The work in [18] showed that there exists a large computational overhead with the Python implementation of LEDBAT algorithm resulting in underutilizing the available network bandwidth more than TCP [18]. Our previous work in [6] analysed the impact of different values of gain on LEDBAT throughput and fairness. Based on the analysis, we proposed a dynamic gain algorithm for stabilising LEDBAT sending rate.

Although several works have analysed the performance of LEDBAT, to the best of our knowledge no work has studied the impact of route changes on LEDBAT throughput and fairness. This is unlike the works in [8], [9] that studied the impact of route changes on the performance of TCP-Vegas. The authors of [9] showed via packet level simulations that route changes resulting in an increasing path delay severely affect TCP-Vegas throughput. To solve the problem, they proposed the use of any lasting increase in the RTT as an indication of route changes. Compared to the solution proposed in [9], the solution in [8] does not require any critical parameter value.

In this paper, we make an effort to fill this gap and quantify the impact of route changes on the throughput and fairness of LEDBAT. Our focus is on the effect of the average time between successive changes of path delay, average magnitude of change of path delay, as well as the effect of increasing and decreasing the path delay.

III. Scenario Description and Assumptions

Our network scenario is based on Figure 1. This represents an ISP network with multiple customers representing traffic sources and sending data to various destinations via a

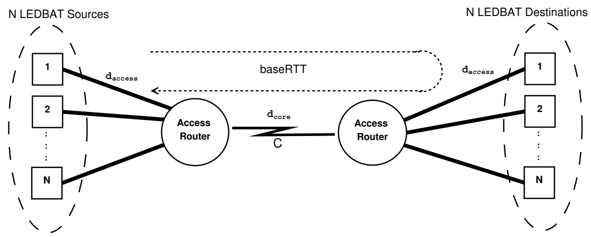

Figure 1. Network Topology

bottleneck link. The sources run file sharing applications exploiting LEDBAT congestion control algorithm. For the values of LEDBAT design parameters, we use 25ms and 40 for d_{tar} and G, respectively [1]. The link delay between the access routers is d_{core}, i.e. "AccessRouter—AccessRouter" shown in Figure 1. Although d_{core} is assigned to the bottleneck link in Figure 1, it in fact represents the delay across multiple links.

The following are the assumptions made for this analysis:

- As LEDBAT throughput when co-existing with other traffic sources will be low most times, other sources have finished their session and only LEDBAT traffic is present in the access network.
- The uplink from the access router to the next router is the bottleneck link in the path for end users, a typical case in most ISP networks [3], [1]. The uplink has capacity C. Therefore, the capacities of all other links are greater than C.
- As access routers in most ISP networks lack AQM [1], the router uses a FIFO drop-tail queue with maximum size of B packets.
- The capacity of each link across every new route beyond the ISP network is no less than C. This is because the capacities of most links in the Internet are usually in the order of gigabits or several hundreds of megabits [19], [20]. Therefore, C still remains the bottleneck after route changes.
- As a possible consequence of route changes in the Internet is a significant change in the path delay of the new route, d_{core} in Figure 1 and hence d_{path} are not fixed but vary when LEDBAT is already in steady state[2].
- The path one-way delay from the source to destination is the summation of all delays in the path and denoted as d_{path}. That is, $d_{path} = d_{core} + d_{access} + d_{que}$.

IV. Modelling LEDBAT Congestion Window When Route Changes

LEDBAT congestion window can limit the throughput of a LEDBAT source [4]. To provide a formal explanation of the behaviour of LEDBAT, we will present in this section a model of LEDBAT congestion window when route changes. Here and in Section V-D, we use Δd_{path} in a general sense to describe the case of increasing or decreasing path delay. Two cases are considered. First is when $\Delta d_{path} < 0$ while the second is when $\Delta d_{path} > 0$, where the path delay of the old route (d^{old}_{path}), the path delay of the new route (d^{new}_{path}), and Δd_{path} are related as $\Delta d_{path} = d^{new}_{path} - d^{old}_{path}$. To mathematically express LEDBAT congestion window, the next section formally describes how a LEDBAT source updates the measured one-way delays. The definitions of other symbols used in this paper are given in Table I.

[2]LEDBAT is in steady state when it has fully saturated the bottleneck capacity and queue delay is approximately equal to the LEDBAT target delay.

Table I
Symbols used with their respective definitions

Symbols	Definitions
d^b	Base one-way delay
d^c	Current one-way delay
$\mathbf{D^b}$	A set of base one-way delays
$\mathbf{D^c}$	A set of current one-way delays
t_{update}	Update interval of $\mathbf{D^b}$
n	Size of $\mathbf{D^b}$
Δd^{ave}_{path}	Average magnitude of change of d_{path}
t^{ave}_{change}	Average time between successive changes of d_{path}
m	Size of $\mathbf{D^c}$

A. LEDBAT Source Updating One-way Delays

A formal description of how a LEDBAT source updates the lists of base and current one-way delays is presented in this section. This will be used in mathematically explaining the behaviour of LEDBAT congestion window when route changes.

A LEDBAT source maintains a set of minimum one-way delays updated every t_{update}, expressed as $\mathbf{D^b} = \{d^b_1, d^b_2, \ldots, d^b_n\}$ where $d^b_1, d^b_2, \ldots, d^b_n$ are previous base one-way delays observed and n is bounded by $2 \leq n \leq n_{max}$. By default, t_{update} is 60 seconds and the maximum size of $\mathbf{D^b}$ (n_{max}) is 10 [1]. If $n = n_{max}$ and the time t_{update} has elapsed, the earliest delay (d^b_1) in $\mathbf{D^b}$ is discarded in order to allow the inclusion of the latest delay in $\mathbf{D^b}$. However, for every one-way delay measured by the LEDBAT source, if the time t_{update} has not elapsed, d^b_n is re-computed according to:

$$d^b_n = \begin{cases} d^c & \text{if } d^c < d^b_n \\ d^b_n & \text{otherwise} \end{cases} \tag{2}$$

The source updates its congestion window w using a base one-way, d^b_{min}, which is the minimum base one-way delay from previous observations, i.e. $d^b_{min} = \min\left(\mathbf{D^b}\right)$.

The LEDBAT source also maintains a set of current one-way delays updated every time an ACK is received. Similar to $\mathbf{D^b}$, the set can be expressed as $\mathbf{D^c} = \{d^c_1, d^c_2, \ldots, d^c_m\}$ where $d^c_1, d^c_2, \ldots, d^c_m$ are previous current one-way delays observed and m is bounded by $1 \leq m \leq w(t)/2$. If $m =$

$w(t)/2$ and a new one-way delay is measured, the earliest one-way delay (d_1^c) is removed from $\mathbf{D^c}$ in order to allow $\mathbf{D^c}$ to be updated. A minimum current one-way delay (d_{min}^c), obtained from taking the minimum one-way delay in $\mathbf{D^c}$, is also used by the source in updating its congestion window. That is, $d_{min}^c = \min(\mathbf{D^c})$.

B. Case 1 of when $\Delta d_{path} < 0$

In this case, we will re-write the equation of LEDBAT congestion window in (1) when no packet loss, considering the impact of a change in d_{path} such that d_{path}^{new} is less than d_{path}^{old}.

Note that $d_{min}^b = d_{path}$ and LEDBAT is already in steady state before the change in d_{path}. This means that the bottleneck link is already saturated and LEDBAT always has packets in the bottleneck buffer. After the change in d_{path}, each of d_i^b in $\mathbf{D^b}$ is assumed to be greater than d_{path}^{new} plus the actual queue delay (d_{que}) where $i = 1, ..., n$. This is because $\Delta d_{path} < 0$. Upon receiving an ACK, the minimum base delay becomes the new path delay plus queue delay, i.e. $d_{min}^b = d_n^b = d_{path}^{new} + d_{que}$ (see (2)). Similarly, $d_{min}^c = d_{path}^{new} + d_{que}$ because $\mathbf{D^c}$ is updated for every ACK received. The source will estimate the queue delay to be zero because $d_{min}^b = d_{min}^c$. Re-writing (1) when no packet loss, we have the following equation representing the maximum increase of w when $\hat{d}_{que} = 0$:

$$w(t+1) = w(t) + G\frac{d_{tar}}{w(t)} \tag{3}$$

Equation 3 shows that the LEDBAT source will assume the access router queue is empty and increase its sending rate until $\hat{d}_{que} \geq d_{tar}$. In effect an additional average queue delay of approximately d_{tar} is caused by the LEDBAT source in steady state. Although this does not affect the average LEDBAT throughput, the objective of keeping queue delay as low as the target delay is compromised resulting in an additional waiting time for newly arriving real-time traffic and possibly impairing the performance of voice, video, and game applications. Equation 3 also shows that the level of impact of a change in d_{path} such that $\Delta d_{path} < 0$ is independent of the magnitude of Δd_{path} instead it depends on the target delay d_{tar}.

C. Case 2 of when $\Delta d_{path} > 0$

We now consider another case where d_{path}^{new} is greater than d_{path}^{old} and re-write (1) when no packet loss, considering the impact of the change in d_{path}.

As in Case 1, $d_{min}^b = d_{path}$ and LEDBAT is already in steady state before the change in d_{path}. After the change in d_{path}, each of d_i^b in $\mathbf{D^b}$ is assumed to be less than d_{path}^{new} because $\Delta d_{path} > 0$. Increasing the path delay increases the bandwidth-delay product. When the path delay increases, the number of packets in the queue decreases. That is, more packets (excluding queued packets) in transit. Upon the change in d_{path} and an ACK packet is received, d_n^b remains unchanged before t_{update} elapses. Even after a time of t_{update} and $\mathbf{D^b}$ is updated, d_{min}^b remains unchanged until approximately a time of $t_{update} \times n$. As a result, $d_{min}^b = d_{path}^{old}$. As $\mathbf{D^c}$ is updated every ACK received and assuming a relatively small size of $\mathbf{D^c}$ depending on the steady state congestion window of LEDBAT before the change in d_{path} occurs, $\mathbf{D^c}$ is filled with the new one-way delays such that $d_{min}^c = d_{path}^{new}$.

Ideally, $d_{min}^b = d_{path}^{old} + \Delta d_{path} = d_{path}^{new}$ indicating that the LEDBAT source has correctly estimated the base one-way delay of the new route. The correctly estimated queue delay is the ideal value and in this case it will be zero as $d_{min}^c = d_{path}^{new}$. As a result, the LEDBAT source congestion window would have been updated using (3). However, the source does the opposite and the actual estimated queue delay of the new route is simply the change in the path one-way delay, Δd_{path}. We therefore re-write (1) when no packet loss as:

$$w(t+1) = w(t) + G\frac{d_{tar} - \Delta d_{path}}{w(t)} \tag{4}$$

From (4), it can be inferred that the magnitude of the change in the path one-way delay can limit the LEDBAT throughput as $w(t+1) < w(t)$ if $\Delta d_{path} > d_{tar}$. As d_{min}^b and hence d_{que} remain incorrectly estimated for a time of $t_{update} \times n$, $w(t+1)$ will always be less than $w(t)$ for the same period of time resulting in an average access router queue delay less than the LEDBAT target delay d_{tar}.

V. Performance Analysis

In this section we present simulation results showing the negative impacts of route changes on LEDBAT throughput (of fully utilizing the bottleneck when no traffic exists) and fairness (of keeping queue delay as low as the target delay) under different conditions. Key performance metrics are LEDBAT congestion window, access router queue delay, and LEDBAT throughput. Note that where we report the normalized throughput we mean the ratio of the actual throughput to the ideal throughput.

A. Simulation Setup

The scenario described in Section III was simulated in ns-2.34 [21] using a similar topology to Figure 1. We used our implementation of the detailed LEDBAT algorithm in ns-2.34 [6]. Every LEDBAT flow will respond accordingly to changes in delay in the path and intra-protocol fairness issues of LEDBAT have been analysed in [16], [17], [5]. Therefore, we consider a single LEDBAT source to simplify the exposition. In the simulations, $C = 2$Mb/s and $B = 100$ packets while all other links are set to 100Mb/s capacity and 5ms link delay. We used these values for C and the capacity of other links because we want C to remain the bottleneck. In all simulations, the default values of t_{update}

and n_{max} are used. The following scenarios are considered in our simulations:

Scenario 1: LEDBAT starts at time 0 and stops at time 60s. Path delay begins to vary at time 30s till the end of LEDBAT session.

Scenario 2: LEDBAT starts at time 0 and stops at 900s. Path delay changes once at time 120s.

Scenario 3: LEDBAT starts at time 0 and runs for 480s. Path delay begins to vary at 30s till the end of LEDBAT session.

Scenario 4: LEDBAT starts at time 0 and runs for 480s. Path delay changes once at time 30s.

Based on studies of delay in the Internet [22], [23], [24], we model varying path delay using two variables each chosen from independent exponential distributions: the average magnitude of change of path delay (Δd_{path}^{ave}) and the average time between successive changes of path delay (t_{change}^{ave}).

Scenarios 1 and 3 consider the impact of Δd_{path}^{ave} and t_{change}^{ave}. For this case, the value of d_{core} is set to 25ms such that the new average value of d_{path} over a period of time is no less than 25ms. We consider 30, 60, and 90 milliseconds as the average magnitude of change of path delay. t_{change}^{ave} is varied from 1 to 50 seconds with a default value of 1 second. In the LEDBAT algorithm, $\mathbf{D^b}$ is normally updated every t_{update}. We consider the rate at which route changes occur to be less than every t_{update}. The case of when t_{change}^{ave} is greater than t_{update} is not considered because we expect that the LEDBAT source to quickly detect such a change and estimate the correct base one-way delay. Therefore, different values of t_{change}^{ave} less than t_{update} are used with default value of 1s.

Scenarios 2 and 4 consider the impact of increasing and decreasing the path one-way delay for at most once during the LEDBAT session. For these scenarios, d_{core} is set to 120ms in order to investigate a large decrease in d_{path}. Although instantaneous values were collected for Scenario 2, we allowed the simulations to last for 900s compared to Scenario 1 because we are interested in showing the behaviour of LEDBAT congestion window and the access router queue delay during a period of $n_{max} \times t_{update}$. This is a period after which a LEDBAT source maintains a new set of measured one-way delays. The value of d_{core}, and hence d_{path}, change once at time 120s.

B. Congestion Window and Queue Delay Over Time

Using Scenario 1, Figure 2 shows the time evolution of LEDBAT congestion window and the access router queue delay with different values of Δd_{path}^{ave}. Before d_{path} begins to vary at time 30s, LEDBAT increases its congestion window until it estimates the queue delay to be approximately equal to a target delay of 25ms where it reaches steady state and remains there until a time of 30s. At time 30s, d_{path} starts to vary for different values of Δd_{path}^{ave} (30, 60, and 90 milliseconds). The results shown in Figure 2 illustrate how LEDBAT responds to changes in the path one-way delay every 1s. Additionally, they show the increasing negative impact on LEDBAT congestion window as we increase the average magnitude of change in the path delay (Δd_{path}^{ave}) from 30 to 60 milliseconds. LEDBAT behaves this way because of incorrect estimation of the base one-way delay within the time of t_{change}^{ave}. As it will be shown later in Section V-C, this can lead to underutilization of the available bottleneck capacity, compromising the objective of saturating the bottleneck link when no traffic exists.

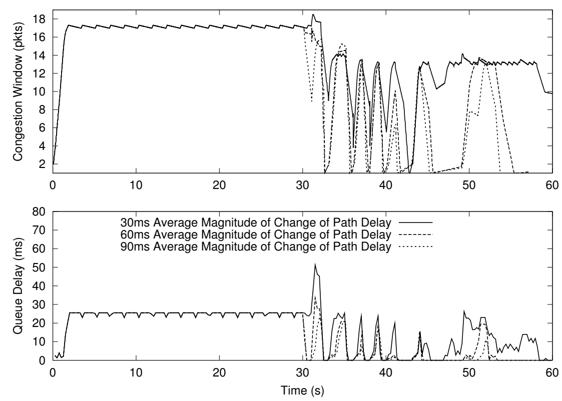

Figure 2. LEDBAT congestion window and the access router queue delay for different average magnitude of change of path delay (Δd_{path}^{ave})

Figure 3 shows the behaviour of LEDBAT congestion window and the access router queue delay over time when the delay of the new path is less than that of the old path, i.e. $\Delta d_{path} < 0$, for Scenario 2. In this case, the change in d_{path} is fixed and occurs once at time 120s. Although LEDBAT detects that d_{path} has changed at 120s as indicated by the changes in the trend of the curves of its congestion window and access router queue delay, an additional queue delay of approximately 25ms is induced by the LEDBAT source (see (3)). This does not affect the throughput for the LEDBAT source but results in an extra queue delay of approximately target delay, thus not meeting the objective of keeping queue delay as low as the target delay. Upon the arrival of traffic from low-latency tolerant applications in the same access network, the additional queue delay may degrade the performance of such applications. At time 840s, the queue delay is increased for an additional value of approximately target delay. This is because the access router queue is never empty to allow the correct estimation of the base one-way delay of the new route, even when $\mathbf{D^b}$ contains a new set of one-way delays.

It has been shown that the fairness objective of LEDBAT of keeping queue delay as low as the target delay may not be met for the change in d_{path} such that $\Delta d_{path} < 0$. We now present results showing how LEDBAT congestion window reverts to its minimum value of 1 packet due

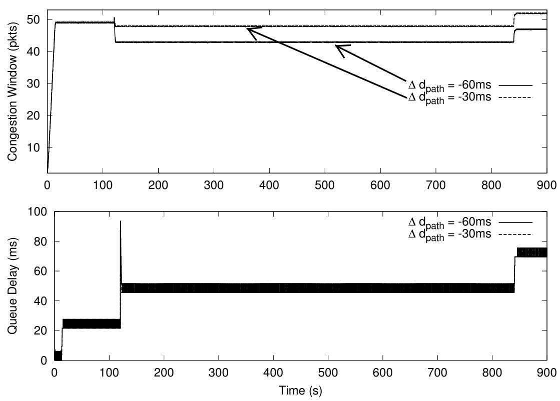

Figure 3. LEDBAT congestion window and the access router queue delay for the case of when Δd_{path} is less than zero and the change in the route is fixed and occurs once

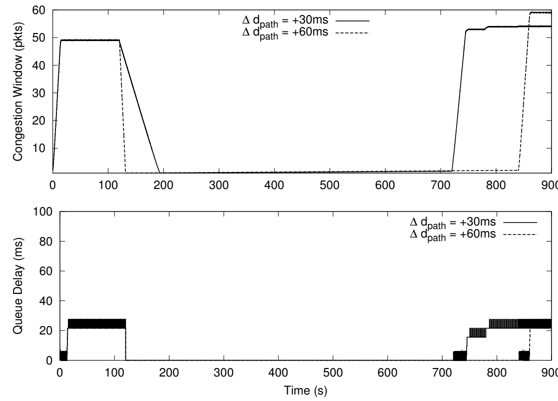

Figure 4. LEDBAT congestion window and the access router queue delay for the case of when Δd_{path} is greater than zero and the change in the route is fixed and occurs once

to wrong base one-way delay estimation by the LEDBAT source for the case of when $\Delta d_{path} > 0$ in Figure 4, causing underutilization of the bottleneck capacity.

Considering Scenario 2, the results given in Figure 4 shows the behaviour of LEDBAT congestion window and the access router queue delay over time for different amount of increasing d_{path}. At time 120s, different magnitude of Δd_{path} causes different decreasing rates of LEDBAT congestion window (see (4)). This results in LEDBAT congestion window reaching its minimum value at approximately 130s for $\Delta d_{path} = +60ms$ and 190s for $\Delta d_{path} = +30ms$. The LEDBAT congestion window remains at the minimum value until the source accurately estimates the base one-way delay at times 720s and 840s. The different values are due to the different changes in d_{path}. Before these times, the queue of the access router is observed to be empty thus allowing the LEDBAT source to accurately measure the new base one-way delay. At the times 720s and 840s, all the previously measured base one-way delays before the change in d_{path} have been popped out from $\mathbf{D^b}$ leaving behind only the base one-way delays measured after the change occurs. An indication that the LEDBAT source has correctly measured the base one-way delay in its path is the access router queue delay observed to increase until it reaches the target delay, including the congestion window as shown in Figure 4. These results validate our discussion in Section IV, showing that the magnitude of Δd_{path} can significantly impact the LEDBAT congestion window and hence throughput.

In addition to the performance of LEDBAT over time, we present results in subsequent sections showing the performance of LEDBAT on the average when route changes.

C. Effect of the Average Time Between Successive Changes of Path Delay (t^{ave}_{change})

For Scenario 3, several simulations are run for 480s using different values of t^{ave}_{change} and Δd^{ave}_{path} for 20 seed numbers. We use the values of Δd^{ave}_{path} to represent the magnitudes of increase in d_{path}. Average and normalized values of LEDBAT throughput, average access router queue delay, and 95% confidence interval of LEDBAT throughput were collected. We started to record all statistics when d_{core} begins to vary at time 30s. The results in Figure 5, Figure 6, and Table II represent average values of 20 seed numbers.

Figure 5 shows the increasing normalized LEDBAT throughput as we increase the average times between successive changes of path delay with different average magnitude of change of path delay (30, 60, and 90 milliseconds) considering Scenario 3. The figure illustrates the increasing negative impact of route changes on the throughput of LEDBAT as Δd^{ave}_{path} increases from 30 to 90 milliseconds. LEDBAT throughput decreases as t^{ave}_{change} decreases. This is due to the frequent decrease and increase of LEDBAT congestion window shown in Figure 2. Figure 6 shows that the average queue delay for 30ms average magnitude of change of path delay is higher than 60 and 90 milliseconds average magnitude of change of path delays. Higher average queue delay means that the LEDBAT source sends more packets. This leads to an increased utilization of the bottleneck link and hence increased throughput shown in Figure 5. In Figure 5, the normalized throughput is observed to be no greater than 0.7. This is due to the fact that the sending rate of a LEDBAT is unstable in the presence of delay variability as LEDBAT congestion window equation depends on delay in a network (see (1) when no packet loss).

Table II shows the increasing average throughput of LEDBAT, including the 95% confidence interval, as the average time between successive changes of path delay (t^{ave}_{change}) increases for each value of the average magnitude of change of path delay (Δd^{ave}_{path}) for Scenario 3. We include the 95% confidence interval of the average LEDBAT throughput in Table II to show the accuracy of our results.

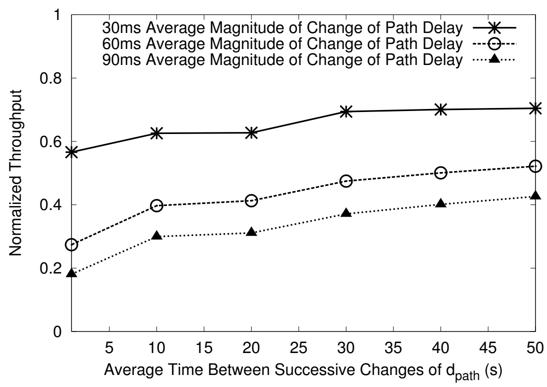

Figure 5. Normalized throughput of LEDBAT for different average magnitude of change of path delay (Δd_{path}^{ave}) and the average time between successive changes of path delay (t_{change}^{ave}) using 20 seed numbers

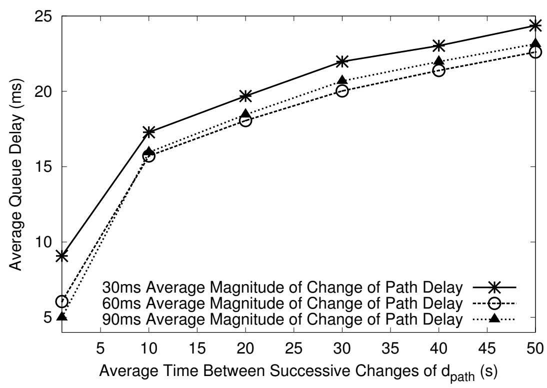

Figure 6. Average queue delay at the access router for different average magnitude of change of path delay (Δd_{path}^{ave}) and the average time between successive changes of path delay (t_{change}^{ave}) using 20 seed numbers

Table II
AVERAGE AND 95% CONFIDENCE INTERVAL OF LEDBAT THROUGHPUT FOR DIFFERENT AVERAGE MAGNITUDE OF CHANGE OF PATH DELAY (Δd_{path}^{ave}) AND THE AVERAGE TIME BETWEEN SUCCESSIVE CHANGES OF PATH DELAY (t_{change}^{ave}) USING 20 SEED NUMBERS.

Δd_{path}^{ave} (ms)	t_{change}^{ave} (s)	LEDBAT Throughput (Kb/s)	
		Average	**95% Conf Interval**
30	1	1132.07	26.4727
	10	1251.3	62.1605
	20	1254.62	75.8655
	30	1388.63	98.2036
	40	1401.34	98.2207
	50	1409	107.673
60	1	548.087	17.1893
	10	794.383	33.9203
	20	825.614	58.4188
	30	949.374	81.5401
	40	1001.57	102.897
	50	1043.5	107.28
90	1	361.411	13.4631
	10	599.128	37.5438
	20	622.239	62.2093
	30	742.398	70.9021
	40	802.369	109.14
	50	852.302	124.786

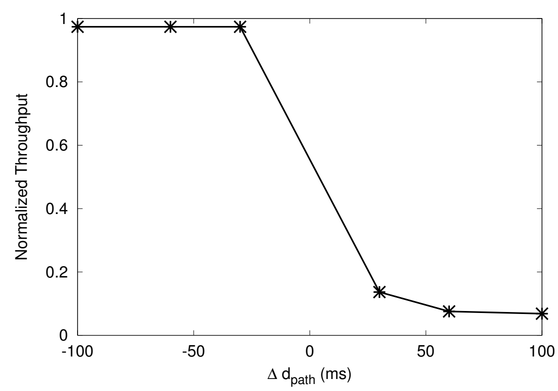

Figure 7. Normalized throughput of LEDBAT for different amount of decreasing and increasing the path one-way delay across a route

D. Effect of Decreasing and Increasing d_{path}

We now present results for Scenario 4 using different values of Δd_{path}. Route changes once at time 30s and the normalized LEDBAT throughput and average access router queue delay are recorded from 30s to 480s for each value of Δd_{path}. Here we consider a fixed change in d_{path} and not average from exponential distribution.

Using Scenario 4, the results in Figure 7 show that LEDBAT throughput is unaffected by any change in d_{path} such that $\Delta d_{path} < 0$. However, as Δd_{path} increases beyond zero, the throughput is observed to decline to a value that is less than 20% of the ideal throughput of approximately 2Mb/s. This is due to the incorrect estimation of the base one-way delay by the LEDBAT source when the change in d_{path} occurs. The incorrect estimation of the base one-way delay leads LEDBAT to unduly decrease its congestion window when the change in d_{path} is greater than the target delay (see (4)).

Although LEDBAT throughput is unaffected when $\Delta d_{path} < 0$, results in Figure 8 show that additional queue delay at the access router is caused by the source. This can lead to more waiting time in the access router queue for newly arriving traffic generated by low-delay tolerant applications. This is due to the wrong base one-way delay estimation by the LEDBAT source (see Section IV). For the case of when $\Delta d_{path} > 0$, the results in Figure 8 also show that the decreasing LEDBAT throughput caused by Δd_{path} increasing beyond zero in Figure 7 is as a result of the average queue delay at the access router less than the target delay.

VI. CONCLUSION

This paper has analysed the impact of delay variability due to route changes on the performance of LEDBAT. We give a formal explanation of the behaviour of LEDBAT

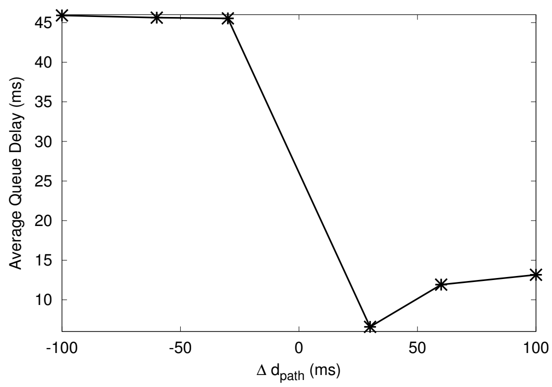

Figure 8. Average queue delay at the access router for different amount of decreasing and increasing the path one-way delay across a route

congestion window when route changes and later consider the effects of the average time between successive changes of path delay and decreasing/increasing path delay across the new route. Our results from several simulations show the negative impact of route changes on the performance of LEDBAT in terms of throughput for a LEDBAT source and fairness with other sources due to incorrect estimation of base one-way delay by the LEDBAT source. In effect, the key LEDBAT objectives of fully utilizing the bottleneck capacity when no traffic exists and of keeping queue delay as low as the target delay may not always be met especially when route changes. In future, we will develop techniques that can make LEDBAT more robust and responsive to route changes thus still meeting the key objectives of throughput and fairness.

REFERENCES

[1] S. Shalunov, "Low Extra Delay Background Transport (LEDBAT)," IETF Internet Draft, (work-in-progress), Mar. 2010, http://tools.ietf.org/pdf/draft-ietf-ledbat-congestion-00.pdf.

[2] uTorrent, "Micro transport protocol (UTP)," http://www.utorrent.com/documentation/utp, accessed on 24th Sep. 2010.

[3] A. Akella, S. Seshan, and A. Shaikh, "An empirical evaluation of wide-area Internet bottlenecks," in *Proceedings of the Intl. Conference on Measurement and Modeling of Computer Systems*, San Diego, CA, 27–29 Oct. 2003, pp. 316–317.

[4] F. Kelly, "Mathematical modelling of the Internet," in *Proceedings of the Fourth International Congress on Industrial and Applied Mathematics*, Edinburgh, Scotland, 5–9 Jul. 1999, pp. 105–116.

[5] G. Carofiglio, L. Muscariello, D. Rossi, and C. Testa, "A hands-on assessment of transport protocols with lower than best effort priority," in *Proceedings of the 35th IEEE LCN*, Denver, CO, Oct. 11–14 2010, p. (to appear).

[6] A. J. Abu and S. Gordon, "A dynamic algorithm for stabilising LEDBAT congestion window," in *Proceedings of the Second International Conference on Computer and Network Technology*, Bangkok, Thailand, Apr. 2010, pp. 157–161.

[7] P. Barford and M. Crovella, "Critical path analysis of TCP transactions," *IEEE/ACM Transaction on Networking*, vol. 9, no. 3, pp. 238–248, Jun. 2001.

[8] K. Srijith, L. Jacob, and A. Ananda, "TCP Vegas-A: Improving the performance of TCP Vegas," *Computer Communications*, vol. 28, no. 4, pp. 429–440, Mar. 2005.

[9] R. J. La, J. Walrand, and V. Anantharam, "Issues in TCP Vegas," Electrical Engineering and Computer Sciences, University of California, Berkeley, Tech. Rep., 1999, available at http://www.eecs.berkeley.edu/~ananth/1999-2001/Richard/IssuesInTCPVegas.pdf.

[10] V. Jacobson and M. J. Karels, "Congestion avoidance and control," *Computer Communication Review*, vol. 18, no. 4, pp. 314–329, Aug. 1988.

[11] L. S. Brakmo and L. L. Peterson, "TCP Vegas: End to end congestion avoidance on a global Internet," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, pp. 1465–1480, Oct. 1995.

[12] A. Venkataramani, R. Kokku, and M. Dahlin, "TCP Nice: A mechanism for background transfers," in *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation*, Boston, MA, 9–11 Dec. 2002, pp. 329–343.

[13] A. Kuzmanovic and E. W. Knightly, "TCP-LP: Low-priority service via end-point congestion control," *IEEE/ACM Trans. on Networking*, vol. 14, no. 4, pp. 739–752, Aug. 2006.

[14] M. Welzl, "A survey of lower-than-best effort transport protocols," IETF Internet Draft, (work-in-progress), Mar. 2010, http://tools.ietf.org/pdf/draft-ietf-ledbat-survey-00.pdf.

[15] D. Rossi, C. Testa, and S. Valenti, "Yes, we LEDBAT: Playing with the new BitTorrent congestion control algorithm," in *Proceedings of the 7th International Conference on Passive and Active Measurement (PAM)*, Zurich, Switzerland, Apr. 7–9 2010, pp. 31–40.

[16] D. Rossi, C. Testa, S. Valenti, and L. Muscariello, "LEDBAT: the new BitTorrent congestion control protocol," in *Proceedings of the 19th International Conference on Computer Communication Networks*, Zurich, Switzerland, Aug. 2–5 2010, pp. 1–6.

[17] G. Carofiglio, L. Muscariello, D. Rossi, and S. Valenti, "The quest for LEDBAT fairness," in *Proceedings of IEEE Globecom*, Miami, FL, Dec. 6–10 2010, p. (to appear).

[18] M. I. Andreica, N. Tapus, and P. Johan, "Performance evaluation of a Python implementation of the new LEDBAT congestion control algorithm," in *Proceedings of IEEE International Conference on Automation, Quality and Testing Robotics*, Cluj-Napoca, Romania, May 28–30 2010, pp. 1–6.

[19] W. Li, Y. Gong, Y. Xu, K. Zheng, and B. Liu, "A 320-Gb/s IP router with QoS control," in *Proceedings of the 15th International Conference on Computer Communication*, Mumbai, Maharashtra, India, Aug. 2002, pp. 134–143.

[20] S. Muller, D. E. J. Gentry, J. E. Watkins, and L. T. Cheng, "High performance network interface," Sep. 17 2002, US Patent 6,453,360.

[21] ns-2, "Network Simulator," http://www.isi.edu/nsnam.

[22] Z. Bo, N. T. S. Eugene, N. Animesh, R. Rudolf, D. Peter, and W. Guohui, "Measurement based analysis, modeling, and synthesis of the Internet delay space," in *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. Rio de Janeiro, Brazil: ACM, Oct. 25–27 2006, pp. 85–98.

[23] A. Corlett, D. I. Pullin, and S. Sargood, "Statistics of one-way Internet packet delays," 2002, available at http://www.ietf.org/proceedings/53/slides/ippm-4.pdf.

[24] K. Kobayashi and T. Katayama, "Analysis and evaluation of packet delay variance in the Internet," *IEICE Transaction on Communications*, vol. 85, no. 1, pp. 35–42, Jan. 2002.