Performance Evaluation of Big Data Transmission Models

Adam H. Villa
Providence College
avilla@providence.edu

Elizabeth Varki
University of New Hampshire
varki@cs.unh.edu

Abstract

Transferring massive data sets across shared communication links are non-trivial tasks that require significant resources and coordination. There are currently two main approaches for servicing these types of big data transmissions: end-to-end and store-and-forward. In end-to-end, one or more data streams are opened between the sender and receiver, and data are transmitted directly over these links. In store-and-forward, data are transmitted from the sender to one or more intermediate nodes, before being forwarded to the receiver. This paper explains these main approaches and identifies the key input parameters for big data transmissions. The paper also provides methods for calculating performance bounds for both end-to-end and store-and-forward approaches. The bounding calculation computes the shortest time by which big data can be transmitted between two locations that may potentially span the globe, which introduces additional complexity. Current research in this area focuses on selecting the optimal routing paths and ideal storage node locations. Since these optimizations are complex and expensive, the computationally cheap bounding techniques presented in this paper can be used to quickly obtain performance estimates.

1 Introduction

Big data transmission refers to the transmission of very large files via the Internet. The file sizes are in the tens of Gigabyte to the Terabyte range. With the proliferation of big data, bulk transmissions now encompass transmissions of Petabyte data sets. Big data or bulk transmissions first gained prominence when the LHC project started; the project was expected to generate petabytes of data that had to be transmitted to researchers around the globe [4]. The LHC project is now on-line with its data sets being transmitted on private high-speed optical links. If a research lab is not directly linked to the high-speed network, then the lab has to rely on the internet or postal mail. Transmitting very large files via the internet is a challenging problem, and is the topic of this paper.

The internet is the largest, most pervasive and diverse distributed system. Each part of the internet is owned and managed by independent institutions. Consequently, the politics and economics of the internet is complex and varied. This is matched by the complexity and variety of the internet's users. Each user competes for bandwidth. Big data transmission requires a disproportionately large fraction of the internet's bandwidth, and this requirement is the essence of the challenge of transmitting large files via the internet. The internet is a shared resource; grabbing a large portion of the bandwidth would negatively impact other users and could eventually crash the network.

The current approach to bulk transmission incorporates the shared nature of the internet and the high bandwidth requirement of bulk transmissions. The options are *a)* to build private high speed networks this approach is followed by LHC to top tier sites; or *b)* to purchase or lease bandwidth on the internet - this approach is followed by internet content producers such as Facebook, Google [1]; or *c)* to delay the transmission of big data until bandwidth is available. This paper focuses on the last approach - delaying transmission until bandwidth is available - referred to as delay tolerant bulk transmissions [12].

Bulk transmissions have received attention in recent years since file sizes have been increasing across-the-board. In addition to cloud players like Amazon, Microsoft, Google, Yahoo!, Akamai, and Facebook, research labs, universities, and even ordinary users are generating larger files. For organizations and labs that transmit large files on a regular basis, it is cost effective to purchase or lease network links. However, ordinary users - in homes, offices, and schools - may also occasionally want to transmit a larger than normal file. In these cases, one has to rely on the internet or postal mail for transmission. Since the need for large file transfers by ordinary users has been growing, there has been substantial interest in delay tolerant bulk transmissions over the internet. The "delay tolerance" refers to the transmissions being given lower priority than regular internet traffic. To ensure that other internet users are not affected, bulk transmissions only grab high bandwidth during periods of low internet usage.

The related papers in the area of delay tolerant bulk transmission protocols can be broadly classified into two, namely, end-to-end protocols - *Ends* - and store-and-forward protocols - *Store*. Ends refers to the TCP/IP approach of transmitting directly from sender to receiver where the speed of transfer is dependent on the smallest link. Bandwidth availability is a function of time; during certain times the network is heavily used, while it is largely idle during other times [10]. This characteristic is used by the Store technique; Store tries to circumvent the smallest link bottleneck by transmitting only when bandwidth is available. Since the sender and receiver may have high bandwidth availability at different times, the sender transmits to intermediate storage nodes; data are transmitted from the storage nodes to the receiver when bandwidth is available at the receiver.

Research studies on bulk transmissions largely focus on transmission routing algorithms where the objective is to get high bandwidth at low cost [11]. A large file is often divided; each part may follow a different route and be stored en-route waiting for bandwidth, before rejoining at the receiver. The algorithms to determine the best routes are computationally expensive [5]; adding to the complexity is the determination of where to place storage nodes, and when to use them [11]. Before deciding on the best transmission route and best buffer store locations, it is beneficial to quickly compute performance estimates for transmission using both approaches. We developed optimistic latency and throughput models for bulk transmissions via Ends and Store.

This paper identifies key input parameters of the internet and its workload that are relevant to big data transmission. A contribution of the paper is that the input parameter values are easy to obtain; the values are relatively stable while capturing the dynamism of the internet. For a given platform, the model outputs the mean latency (response time) for a file transmitted with Ends and Store. Another contribution of the paper is the computational simplicity of the performance models. The models can be used to compute a quick estimate of the time it would take to transmit a large file between a sender and receiver via the internet.

2 Related Work

The advances in bulk data transmission are in three categories, namely, network hardware, network level protocols, and application level protocols.

Hardware: Organizations that regularly transmit bulk data ensure that they have the hardware resources to do so, either by leasing, purchasing or building their network infrastructure. Consequently, the backbone, regional, and consumer ISPs as well as the internet exchanges (IXs) are constantly upgrading their equipment - links, routers, switching fabrics - to keep up with increasing demands from consumers. The larger IXs can handle more than 800 Gb/s during peak traffic times [1]. The intercontinental bandwidth is increased by laying new undersea cables.

Network protocols: Parallel transmission techniques such as GridFTP [2], BitTorrent [15], and Slurpie [17] get as much bandwidth as possible by opening multiple TCP streams. Another class of bulk transmission algorithms deal with finding the best (least congested) network routes between sender and receiver. The most promising of these approaches is OpenFlow [13], based on software defined routing. Each Autonomous Network System (AS) has a centralized high level routing application that has knowledge and control of traffic in the AS. The packet routing decisions are made by this "routing compiler," not individual routers in the network. Google has incorporated OpenFlow in its private networks. Overlay network [16] is another approach that can be used for bulk data transmission since it allows the application to choose the path from sender to receiver. The third approach to bulk transmissions focuses on the delay tolerance of bulk data. Data packets from bulk transmissions are given a lower priority than regular traffic [3, 20] in techniques based on Ends. Delay incorporating routing algorithms for wireless networks where connections are unreliable are based on Store [7, 9].

Application protocols: The application protocols have the same objective as the network protocols, but the smallest unit considered is the file (file-part if the file is divided), whereas the network protocols deal with packets. Several of the application protocols are based on Store since this approach factors in the bandwidth-time relationship, namely, bandwidth is scarce during the majority of the work day and into late night; bandwidth becomes available in the early morning hours. The basic idea of Store is to use storage nodes as staging buffers to synchronize the bandwidth differential and the high bandwidth availability times between sender and receiver. Prior research has focused on finding the cost optimal route from sender to receiver. For example, Netstitcher [11], takes advantage of already-paid for unused bandwidth in the transit networks along the path from sender to receiver; storage nodes are placed in each transit network. The objective function of Netstitcher is to transfer the maximum data with minimum cost in a given time period. A related Store protocol, iDTT [18], shows that peak traffic and cost are reduced, albeit with a slight increase of latency when compared to protocols based on Ends. The algorithms that find the optimal path are complex [5] for transmission based on Ends or Store, which is expected since the architecture of the internet is distributed and dynamic.

Summarizing, the goal of bulk transmission mechanisms is to avail of high bandwidth, while minimally impacting other users. Bandwidth is accessed by opening one or more transmission links between sender and receiver; the transmissions may occur during low traffic periods and along less congested links in order to minimize the impact on other users. The objective function of algorithms is on finding the optimal low cost, high speed route from sender to receiver, by trying to use "free" already-paid for bandwidth during low traffic times [6, 8, 19].

Motivation for this paper: The bulk transmission protocols can be broadly classified into Ends or Store algorithms. Before finding the optimal path, it is often useful and cost-effective to quickly estimate the best performance that can be achieved using either approach. A prior paper [12] presented cost benefit analysis of Ends and Store by evaluating data from transit ISPs. However, there is no prior work on latency and throughput models of Ends and Store. This paper evaluates these models and develops quick best case latency and throughput bounds for bulk data transmissions with Ends and Store.

3 Platform

This section presents an overview of the platform over which bulk transmissions ply in order to understand what input parameters have a major impact on performance.

Internet configuration: The internet is composed of several Autonomous Systems (AS), each AS is managed by a single Internet Service Provider (ISP). The sender and receiver LANs (*i.e.,* end LANs) are stub AS or multi-homed AS and carry local traffic. Transit AS connect end LANs, and carry both local and transit traffic. The end-user ISPs lease or purchase bandwidth from transit ISPs. Typically, ISPs pay for fixed amount of transit bandwidth determined by peak end-user AS bandwidth usage of the transit AS. Economics and politics determine the bandwidth hand-off deals between the various transit ISPs and IXs. The bulk data transmission problem is the following: transmit bulk data from sender to receiver AS via one or more transit AS.

Once data leave the sender LAN, the data traverse one or more transit AS on its route to the receiver's LAN. Figure 1(a) represents the possible paths of transmission; there are several concurrent transmission channels from sender to receiver. Since each channel is a TCP connection, the transmission rate is determined by the minimum bandwidth along that channel. Figure 1(b) shows the bandwidth given to each transmission channel. The sum total given to the transmission cannot exceed 10 Gb/s, the bottleneck bandwidth from amongst the sender, transit, and receiver AS, shown in Figure 1(c).

LAN configuration: A LAN's configuration is dependent on the organization (users) it serves. A typical configuration could be as follows: a user's computer shares the network bandwidth with other users on the subnet; the traffic from the user's subnet may merge with traffic from other subnets in the same LAN, before it is directed to gateway routers that direct the LAN's traffic to one or more transit AS. The maximum bandwidth out of a LAN is the sum of the bandwidth links to the transit AS. For bulk transmission, the maximum bandwidth is the sender's subnet bandwidth, this may be smaller than the bandwidth out of the sender's AS.

Internet traffic pattern: There is a diurnal traffic pattern of low and high Internet traffic that reflects users' sleep-wake cycle. Figure 2 shows the bandwidth usage at a LAN, during a regular week and on a single day. The weekly traffic distribution is a wave - it is periodic with a period (cycle time) of 1 day. The diurnal traffic pattern is not unique to end-user LANs; this pattern has been observed in transit AS [10] and IX [1]. The significance of wave bandwidth usage is that during the early morning hours of 1:00 AM to about 9:00 AM, there is unused bandwidth that could be used for large file transmissions without impacting other users. The bandwidth usage is relatively stable during 5 minute units [12]. To get a snapshot of the dynamic bandwidth usage during a day, the available bandwidth during each hour can be computed as the average of the 5 minute units or the minimum value during the 5 minute units of each hour. Figure 2B shows the snapshot of bandwidth usage during every hour for an example LAN. Figure 1 only represents the static internet architecture; Figures 2A and 2B represents the dynamic internet.

Summary: In addition to the internet architecture - layout of links and their capacity - the bulk transmission platform consists of the internet users, the workload, the distance between the sender/receiver, and the internet economics/politics. All these aspects impact the bandwidth that can be set aside for delay tolerant bulk transmissions. The LAN architecture shows that the *location* of the sender's node within a LAN is critical to the amount of physical bandwidth that a node can access. The location of the sender and receiver LANs is also important because bulk transmissions have to be routed through one or more transit networks. The *time* at which data transmission is initiated is important since the amount of unused bandwidth is a function of time. Thus, "time and place" are critical to the latency and throughput that can be achieved by bulk transmissions.

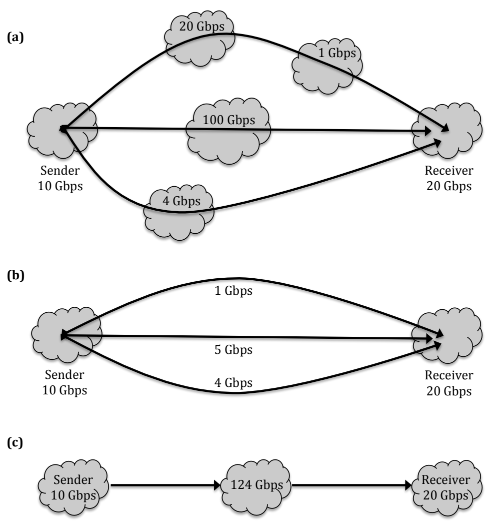


Figure 1: Bulk transmission path from sender to receiver

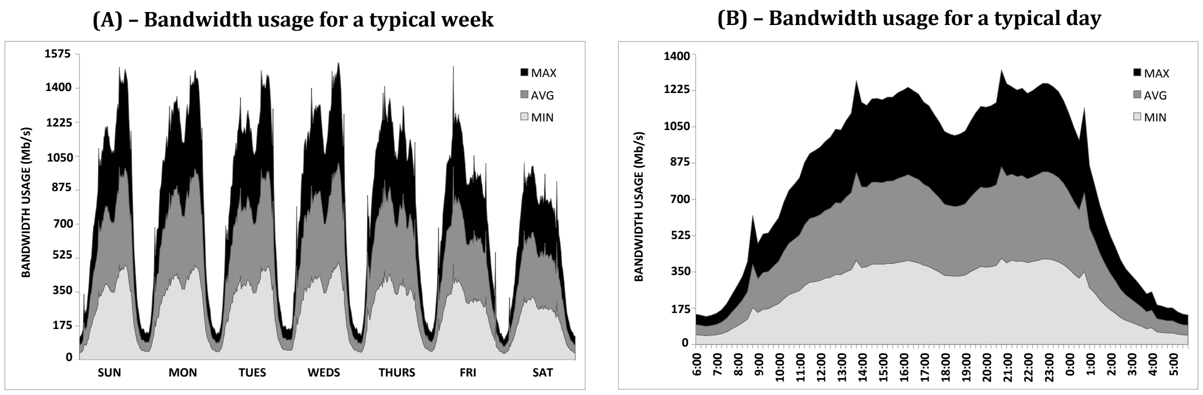


Figure 2: Changes in the minimum, average and maximum bandwidth usage (all receiving and transmitting traffic) for a typical week (A) and typical day (B) for an end-user ISP (LAN).

4 Performance model

From the end user's perspective, the time at which the data completes downloading at the receiver's node is important. From an administrator's perspective, the bandwidth usage and the duration of this usage is of importance. In addition, a user may be interested in ease of use, security, retransmissions, cost, etc. We ignore these aspects since they are related to the application, not the underlying transmission technique.

4.1 Performance metrics

The model computes the following:

1. Transmission Time
   sendTT, recTT$\in \{1,2,...\}$: number of hours that sender and receiver's LANs, respectively, are busy transmitting the file.
   TT$\in \{1,2,...\}$: maximum number of hours that the internet is utilized during the transmission of the file's data.
   Suppose a file is transmitted from sender to an intermediate server in 2 hours; the file is then transmitted to the receiver in 1 hour. In this case, sendTT= 2, recTT= 1, TT= 2, the maximum of the two transmission times.
2. Wait Time
   sendWT$\in \{0,1,2,...\}$: number of idle (no transmission) hours between InitiateTime (time at which user requests bulk transmission) and completion of transmission from the sender's LAN.
   recWT$\in \{0,1,2,...\}$: number of idle (no transmis-

sion) hours at the receiver between InitiateTime and completion of transmission (arrival of file) at the receiver's computer.

3. Response Time
   RT$\in \{1,2,...\}$: number of hours between InitiateTime and the completion of the transmission.

Note that the time unit is an hour - transmissions start on the hour and end on the hour; if a transmission completes in less than an hour, the transmission time is rounded up to an hour.

4.2 Input parameters

The input parameters that assess and differentiate the performances of Ends and Store are identified. In Section 3, time and place are identified as important to the performance of bulk transmissions. We use this information to define the input parameters to the performance model. The traffic pattern has a periodic diurnal pattern, so all parameters pertain to a 24 hour period.

1. Time is encapsulated in the variable InitiateTime which represents the hour at which the user initiates the transmission.
   InitiateTime = j where j $\in \{0,1,2,....,22,23\}$: the hour (time) at which the user initiates the transfer. Hour 0 is 12 AM, hour 1 is 1 AM,..., hour 23 is 11 PM.
2. For bulk data transmission, the relevance of sender and receiver location is captured by 2 input parameters: *bottleneck bandwidth* in sender and receiver AS, and the *time zones* in which the sender and receiver AS are located. The end ISPs typically purchase a fixed bandwidth from the transit ISPs; the end LANs cannot get more bandwidth than the sum of the bandwidth capacity of links to transit AS. Therefore, the bottleneck bandwidth in a user's LAN cannot exceed the bandwidth capacity out of the user's LAN.
   (a) SendBW[i], RecBW[i] represent the smallest available bandwidth (in Mb) from sender's/receiver's node to LAN during hour i; $0 \leq i \leq 23$. The bandwidth availability data are tracked by each ISP, so it is available (but not readily accessible due to restrictions by ISPs). Since this is the bottleneck bandwidth, this is the best performance the bulk transmissions can achieve. The SendBW andRecBW parameters capture the dynamic characteristic of the internet and its users. Note that if the transit bandwidth is lower than either the sender or receiver bandwidth, and this information is provided, then the transit bandwidth can replace the higher of the sender/receiver bandwidth; this input will result in tighter bounds.
   (b) TimeDiff = j where j $\in$ {..., -3, -2, -1, 0, 1, 2, ...} represents the number of hours by which the receiver campus is ahead or behind the sender's campus. For example, if the sender is situated in California and the receiver is situated in Japan, then TimeDiff= 15 (equivalently, TimeDiff= -8). This TimeDiff parameter captures the impact of distance between sender and receiver.
3. Another input parameter of relevance is the size of the data set to be transmitted. Let the file size be represented in GB.
   FileSize =x GB where x $\in \{1,2,....\}$

The input parameter values are easy to obtain; the input parameter values are relatively static, yet they capture the complexity and versatility of the internet and its users.

4.3 Models

The internet is dynamic with hardware, protocols, workload, and traffic upgrades occurring constantly [1]. Each AS is owned and managed by a different ISP; getting performance data from an ISP is not always feasible. All of these factors contribute to the difficulty of evaluating the performance of bulk data transmissions. The transmission may follow several paths from sender to receiver; each path may traverse multiple transit AS. The performance of bulk transmissions depend on the characteristics - bandwidth, storage nodes - of these transit AS.

Instead of trying to model the transit AS, we model the sender AS and the receiver AS. Since the bulk data are transmitted out of the sender's LAN and into the receiver's LAN, these end ASs play a critical role in the performance of bulk transmissions. The transmission capacity cannot exceed the minimum of the sender and receiver's bandwidth capacity. Therefore, using only the sender and receiver's LAN input parameters, we are able to compute the best performance bounds for bulk transmissions between the sender and receiver. However, note that if the transit AS is the bottleneck and this information is provided, then tighter best case bounds can be generated by simply replacing the SendBW values with the bandwidth of the transit AS. For example, in Figure 1, if only the first route is available between sender and receiver, and this information is provided to the model, then the 1 Gb/s AS should be considered as the bottleneck.

4.3.1 Store

The essential difference between Store and Ends is the presence of storage nodes that are used as bandwidth and

Time	Utilization (%)	SendBW/RecvBW
12PM-2AM	95%	15 Mbps
2AM-3AM	50%	500 Mbps
3AM-4AM	20%	800 Mbps
4AM-5AM	15%	850 Mbps
5AM-7AM	10%	900 Mbps
7AM-8AM	15%	850 Mbps
8AM-9AM	20%	800 Mbps
9AM-10AM	35%	650 Mbps
10AM-11AM	45%	550 Mbps
11AM-12PM	65%	350 Mbps

Figure 3: Network utilization and bandwidth availability for each hour of a typical day for an example LAN.

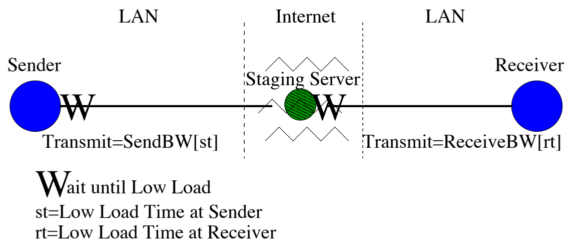


Figure 4: Store

time matching buffers. In order to avail of maximum installed bandwidth without impacting other users, the solution is to open multiple transmission streams during low traffic. If the sender and receiver are in the same time zone then a direct transmission from sender to receiver is feasible. If the sender and receiver are in different time zones, then the low traffic periods at the two end points do not coincide. In this instance, the file is transmitted from the sender to one or more staging server(s), placed in the Internet zone, as shown in Figure 4. Depending on the Internet configuration between the sender and receiver, the file may be transmitted to a single staging server via multiple streams or the file may be divided and parts of the file are transmitted concurrently to multiple staging servers. When the receiver's LAN traffic is low, the file can be transmitted from the staging server(s) to the receiver.

The user initiates the transmission; if it is high traffic at the sender's LAN, then the transmission does not begin until low traffic. At low traffic, the file transmission starts from the sender to the staging server at rate SendBW. If it is currently high traffic at the receiver, the transmitted portions of the file remain at the staging server. When low traffic period starts at the receiver's campus, then transmission proceeds to the receiver. If the sender and receiver are in the same time zone, the staging server can be used to absorb the bandwidth differential between

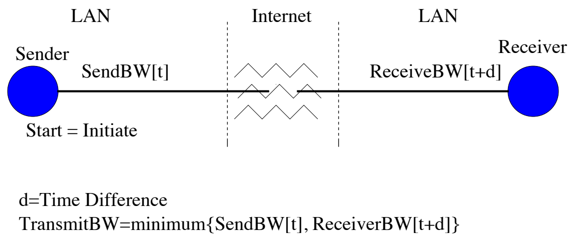


Figure 5: Ends

sender and receiver. For example, if the sender has a bandwidth of 5 Gb/s and the receiver has a bandwidth of 1 Gb/s, the sender can transmit at the higher rate using the staging server as a buffer.

The Store performance model computes best performance estimates for Store. The performance estimates are the best-case bounds since the transmission rate cannot exceed the bottleneck bandwidth at the end LANs. To keep code simple, we have set SendBW[i] = RecBW[i] = 0 when $i \in$ HighTrafficHours. This initialization does not impact performance metrics since Store does not transmit during high traffic hours. However, if the remaining portion of a file is small enough (equivalent to standard files), the code can be modified to allow transmission of this remaining portion of the file during the high traffic time. The variables SendRemain, StageFile, RecFile represent the file sizes at the sender, staging server, and receiver. The variables SendTransmit and RecTransmit represent the amount of data transmitted at the sender and receiver during the current hour. In line 5, the function *TimeZone* computes the time at the receiver given the time at the sender and the difference in time zones between sender and receiver.

4.3.2 Ends

To ensure that other users are not negatively impacted by bulk transmissions, Store waits until low traffic to transmit bulk data. This may be wasteful since there is usually some free (unused) bandwidth (maybe, as small as 10Mb/s) even during high usage periods. Ends, unlike Store, starts transmitting bulk data as soon as the user initiates the bulk transmission; the goal of Ends is to avail of the unused (already purchased) bandwidth instead of waiting until low traffic. Figure 5 depicts Ends transmission of bulk data. The Ends performance model computes best performance. The transmission begins at InitiateTime so sendWT and recWT are 0.

Algorithm 1: STORE PERFORMANCE MODEL

```
Initialize sendTT, recTT, StageFile, RecFile, RT, recWT to 0;
FileNotTransferred= TRUE;
SendRemain= FileSize;
i = InitiateTime;
j = TimeZone(InitiateTime, TimeDiff);
while FileNotTransferred do
    SendTransmit= SendBW[i] * 60 * 60;
    if SendRemain$>$ 0 && SendTransmit$>$ 0 then
        sendTT++;
        if SendTransmit$>$ SendRemain then
            SendTransmit= SendRemain;
        SendRemain= SendRemain- SendTransmit;
        StageFile= StageFile+ SendTransmit;
    RecTransmit= RecBW[j] * 60 * 60;
    if StageFile$>$ 0 && RecTransmit$>$ 0 then
        recTT++;
        if RecTransmit$>$ StageFile then
            RecTransmit= StageFile;
        StageFile= StageFile- RecTransmit;
        RecFile= RecFile+ RecTransmit;
    if (SendRemain$>$ 0 && SendTransmit== 0) || (RecTransmit== 0) then
        recWT++;
    RT++;
    if RecFile$>=$ FileSize then
        FileNotTransferred= FALSE;
    i = (i+1) MOD 24;
    j = (j+1) MOD 24;
TT= MAXIMUM(sendTT, recTT);
print RT, TT;
```

Algorithm 2: ENDS PERFORMANCE MODEL

```
RemainingFile = FileSize;
CompleteTime = 0;
i = InitiateTime;
while RemainingFile $>$ 0 do
    j = TimeZone(i, TimeDiff);
    CurrentBW = MIN(SendBW[i], RecBW[j]);
    TransmittedFile = CurrentBW * 60 * 60;
    RemainingFile= RemainingFile- TransmittedFile;
    CompleteTime = CompleteTime+ 1;
    i = (i + 1) MOD 24;
RT = CompleteTime;
TT = CompleteTime;
```

5 Analysis

This section evaluates the performance models to quantify the impact of InitiateTime, TimeDiff, and the bandwidth differential on performance. The parameters SendBW[i], RecBW[i] capture the impact of traffic intensity on the transmission tool. The notational complexity of the analysis can be simplified, thereby improving the clarity, without changing the essential performance characteristics, by assuming that SendBW and RecBW do not vary by the hour. Instead, the bandwidth available for bulk data transmissions is a fixed low value during the high traffic period and a fixed high value during the low traffic period. Let SendLowBW and SendHighBW represent the low and high transmission rate per hour, respectively, at the sender's LAN; let RecLowBW and RecHighBW represent the low and high transmission rate per hour, respectively, at the receiver's LAN. Note that the transmission rate is given in units of per hour, not per second;

At any hour, the sender and receiver LANs are in one of the following four states: 1) SendLowBW, RecLowBW; 2) SendLowBW, RecHighBW; 3) SendHighBW, RecLowBW; and 4) SendHighBW, RecHighBW. Since end-to-end transmission rate is dependent on the smallest bandwidth, the bandwidth rate at any hour would be one of:

$$\mathsf{Low} = MIN\{\mathsf{SendLowBW}, \mathsf{RecLowBW}\};$$
$$\mathsf{SendLow} = MIN\{\mathsf{SendLowBW}, \mathsf{RecHighBW}\};$$
$$\mathsf{RecLow} = MIN\{\mathsf{SendHighBW}, \mathsf{RecLowBW}\};$$
$$\mathsf{High} = MIN\{\mathsf{SendHighBW}, \mathsf{RecHighBW}\};$$

Let #HighBWHrs and #LowBWHrs represent the number of hours in a day when traffic is off-peak and peak, respectively. For the example LAN presented in Figures 2A and 2B, the off-peak traffic period is from hours 0 to 9, while the peak traffic period is from hours 10 to 23. Therefore, #HighBWHrs = 10 and #LowBWHrs = 14.

5.1 Store

Maximum FileSize transmitted in 24 hours:

Let 24hrFileSize represent the maximum FileSize that can be transmitted during 24 hours.

$$\mathsf{24hrFileSize} = \mathsf{High} \times \mathsf{\#HighBWHrs}$$

Result 1 *For Store, the maximum data that can be transmitted during 24 hours is determined only by* High*;* 24hrFileSize *is independent of* TimeDiff *and* InitiateTime.

Response time RT:

The RT is computed in terms of receiver wait time and receiver transmission time.

$\mathsf{RT} = \mathsf{recWT} + \mathsf{recTT}$

recWT is the time from InitiateTime until the start of transmission to the receiver. The receiver's transmission starts at the first high bandwidth hour that is greater than or equal to the sender's first transmission hour. Since the staging server transmits only when the receiver starts off-peak period, recWT is a function of InitiateTime and TimeDiff.

Result 2 sendWT *is dependent on* InitiateTime. recWT *is dependent on both* InitiateTime *and* TimeDiff. *Consequently,* RT *is dependent on both parameters.*

Transmission times sendTT, recTT, TT:

$\mathsf{TT} = \mathsf{MAX}(\mathsf{sendTT}, \mathsf{recTT}) = \lceil \frac{\mathsf{FileSize}}{\mathsf{High}} \rceil$

The sender's transmission time depends only on the bandwidth at the sender due to the buffering available at the staging servers.

$\mathsf{sendTT} = \lceil \frac{\mathsf{FileSize}}{\mathsf{SendHighBW}} \rceil$

Calculating recTT is tricky; depending on TimeDiff, the transmission from sender to receiver may be completely concurrent, partially concurrent, or serial. The computation of recTT also depends on whether the sender or the receiver is faster.

If $\mathsf{SendHighBW} \geq \mathsf{RecHighBW}$, then

$\mathsf{recTT} = \lceil \frac{\mathsf{FileSize}}{\mathsf{RecHighBW}} \rceil$.

If $\mathsf{SendHighBW} < \mathsf{RecHighBW}$ and $\mathsf{TimeDiff} = 0$, then

$\mathsf{recTT} = \lceil \frac{\mathsf{FileSize}}{\mathsf{SendHighBW}} \rceil$.

If $\mathsf{SendHighBW} < \mathsf{RecHighBW}$ and $|\mathsf{TimeDiff}| > 0$, then the value of recTT depends on how much of the file, StageFile, has been transmitted to the staging server before the receiver's transmission time starts. While the receiver catches up with the sender, the file is transmitted at the receiver's faster rate and then afterward, any remaining portion is transmitted at the sender's slower rate.

$\mathsf{recTT} = \lceil \frac{(\mathsf{StageFile}+)}{\mathsf{RecHighBW}} + \frac{\mathsf{FileSize} - (\mathsf{StageFile}+)}{\mathsf{SendHighBW}} \rceil$.

The + in StageFile+ represents the additional transmission from the sender while the receiver is trying to catch up. As TimeDiff increases, recTT becomes more dependent on RecHighBW (and less dependent on $\mathsf{High} = \mathsf{MIN}(\mathsf{SendHighBW}, \mathsf{RecHighBW})$).

Result 3 sendTT *only depends on* SendHighBW.
As TimeDiff *increases,* recTT *becomes less dependent on* High *and more dependent on* RecHighBW.

The presence of the staging servers ensures that the bandwidth differential between sender and receiver is hidden. Result 3 states that as TimeDiff increases the dependency of RT to SendBWdecreases; the RT becomes more dependent on RecBW.

Result 4 *The* TT *of the Store model only depends on* High*;* TT *is insensitive to* TimeDiff *and* InitiateTime.

5.2 Ends

Maximum FileSize transmitted in 24 hours:

The total data transmitted depends on TimeDiff between sender and receiver.

1) $\mathsf{TimeDiff} = 0$:

$\mathsf{24hrFileSize} = (\mathsf{High} \times \mathsf{\#HighBWHrs}) + (\mathsf{Low} \times \mathsf{\#LowBWHrs})$

2) $|\mathsf{TimeDiff}| = d$ where $0 < d \leq \mathsf{\#HighBWHrs}$:

$\mathsf{24hrFileSize} = (\mathsf{RecLow} \times d) + (\mathsf{High} \times (\mathsf{\#HighBWHrs} - d)) + (\mathsf{SendLow} \times d)) + (\mathsf{Low} \times (\mathsf{\#LowBWHrs} - d))$

3)$|\mathsf{TimeDiff}| = d$ where $\mathsf{\#HighBWHrs} < d \leq \mathsf{\#LowBWHrs}$

$\mathsf{24hrFileSize} = (\mathsf{RecLow} \times \mathsf{\#HighBWHrs}) + (\mathsf{Low} \times (d - \mathsf{\#HighBWHrs})) + (\mathsf{SendLow} \times \mathsf{\#HighBWHrs})) + (\mathsf{Low} \times (\mathsf{\#LowBWHrs} - d))$

From 3), it follows that when transmitting between LANs in India and the US, or between LANs in Japan and the US, the entire Ends transmission is carried out in low bandwidth. Note that $23 \geq d > \mathsf{\#LowBWHrs}$ is not evaluated since it reduces to one of the above cases.

Result 5 *Ends: When* TimeDiff*=0,* 24hrFileSize *is maximum. As* TimeDiff *increases,* 24hrFileSize *decreases. At* $\mathsf{TimeDiff} \geq \mathsf{\#HighBWHrs}$*, the data are entirely transmitted at low bandwidth, so* 24hrFileSize *is smallest.*

When $\mathsf{TimeDiff} = 0$, Ends transmits more data than Store. However, depending on the difference in bandwidth during high traffic and low traffic times, the percentage improvement may be insignificant. For example, for a 10Mb low bandwidth and a 1 Gb high bandwidth, Ends transmits 1.6% more data than Store during the high traffic hours.
For Ends: $\mathsf{RT} = \mathsf{TT} = \mathsf{sendTT} = \mathsf{recTT}$.

Result 6 *The transmission time of Ends is sensitive to parameters* InitiateTime*,* TimeDiff*, bandwidth availability, and transmission rate differential between sender and receiver.*

5.3 Summary

Theorem 1 *For a given* FileSize*, the* TT *of Store is at least as fast as that of Ends.*
Ends is best in comparison to Store when TimeDiff *is close to 0 and the sender and receiver have similar transmission rates.*
Ends is worst in comparison to Store when there is no overlap of low traffic times between sender and receiver.

Thus, Store is better suited to bulk transmissions over large distances that span time zones and varying network capabilities. However, Ends, unlike Store, does not require storage nodes, so Ends may be more cost effective if the dynamic bandwidth differential between sender and receiver is not high. Also, if the sender has substantially more bandwidth than the receiver at all times, then it is cost effective to just wait until the receiver's high bandwidth capacity time and then transmit directly from the sender.

6 Experiments

The goal of the simulations is to verify and validate the theoretical analysis of the last section. OPNET is a commercial simulator capable of simulating a wide variety of network components and workloads [14]. Three client/server machine pairs are setup to emulate the workload of the most popular traffic classes: streaming video, web browsing and VoIP. In the simulations, the workloads of the three popular traffic classes are varied to represent the background utilization of the shared Internet connection at various times of the day. The fourth server is used to simulate big data transfers. Another server handles staging for big transmissions.

TT as TimeDiff and InitiateTime varies:
Figure 6 plots TT as InitiateTime varies; each graph has a different value for TimeDiff. The graphs shows that Ends and Store are closest in performance when InitiateTime lies in a low traffic period and TimeDiff is small. The last graph plots the percentage improvement in transmission time of Store when compared to Ends for each TimeDiff. As TimeDiff increases, the performance of Ends degrades, while the performance of Store remains unchanged. There is a 100% improvement in TT of Store when $\mathsf{TimeDiff} = 12$. This can also be seen in Figure 7 which plots TT as TimeDiff varies over a 24 hour period. When $\mathsf{TimeDiff} = 12$, the sender and receiver are in antipodal time zones (*i.e.,* there is no overlap of low traffic times at the sender and receiver), and the Ends model always transmits over small bandwidth.

RT as TimeDiff and InitiateTime varies:
Figure 8 plots RT graphs. Note that for the Ends model, $\mathsf{RT} = \mathsf{TT}$. Both models perform better when $\mathsf{TimeDiff} = 0$. As TimeDiff increases, in Store the wait time increases; in Ends, the time zones don't synchronize resulting in low bandwidth transmission.

Bandwidth differential between sender and receiver:
Figure 9 plots recTT (transmission time at receiver's LAN) when the receiver has 4 times as much bandwidth as sender. For Ends, the faster transmission rate at receiver has no impact on performance. For Store, when $\mathsf{TimeDiff} = 0$, the faster transmission rate of receiver has no impact; however, when $\mathsf{TimeDiff} > 0$, the recTT is faster. Compare graph 1, graph 3 of Figure 6 with graph 1, graph 2 of Figure 9, respectively.

Summary
The graphs show the impact each input parameter has on the performance of bulk transmissions. The bounding model incorporates parameters from the sender and receiver LANs, not the transit networks. This is fine for the bounds since performance cannot increase beyond the limits of an end LAN's link to the internet. However, it is possible that a LAN's internet link is greater than the bandwidth it can get to the receiver LAN. If this transit bottleneck is identified, then tighter bounds can be generated by setting the sender's bandwidth to the bottleneck transit bandwidth.

7 Advantages

The paper has developed performance models for Ends and Store, the two fundamental approaches to delay tolerant bulk transmissions. The models output optimistic bounds on performance when bulk data are transmitted via the internet. The advantages of using this bounding technique to evaluate big data transmissions are:

1. the input parameters required by the bounding technique are easily available. Even if the end user ISPs do not provide bandwidth usage data, the maximum physical bandwidth of the user's link can be used to compute bounds. Moreover, the input parameter values are relatively static, yet incorporate the versatility of the internet.
2. the bounding technique does not require data from transit networks. Transit ISPs rarely disclose information about their networks; moreover, transit networks are constantly being upgraded to keep up with demands. Therefore, bypassing the transit networks is an important benefit.
3. the bounds can be used as a yardstick to evaluate and compare the performances of bulk transmission routing algorithms. A bulk transmission routing algorithm based on Store should be compared against the Store bound, while an algorithm based on Ends should be compared against the Ends bound. Currently, papers that propose new algorithms compare their techniques against other transmission techniques. However, if a Store algorithm is compared against an Ends algorithm, then it is difficult to extract performance gains from routing decisions. For example, if Netstitcher [11], based on Store, is compared to BitTorrent, based on Ends, it is hard to gauge whether the performance gain is due to routing decisions or due to Store vs. Ends disparity.
4. Section 5 showcases the essential features of Ends and Store. It is easy to identify the platform over which one transmission technique is superior to the other.

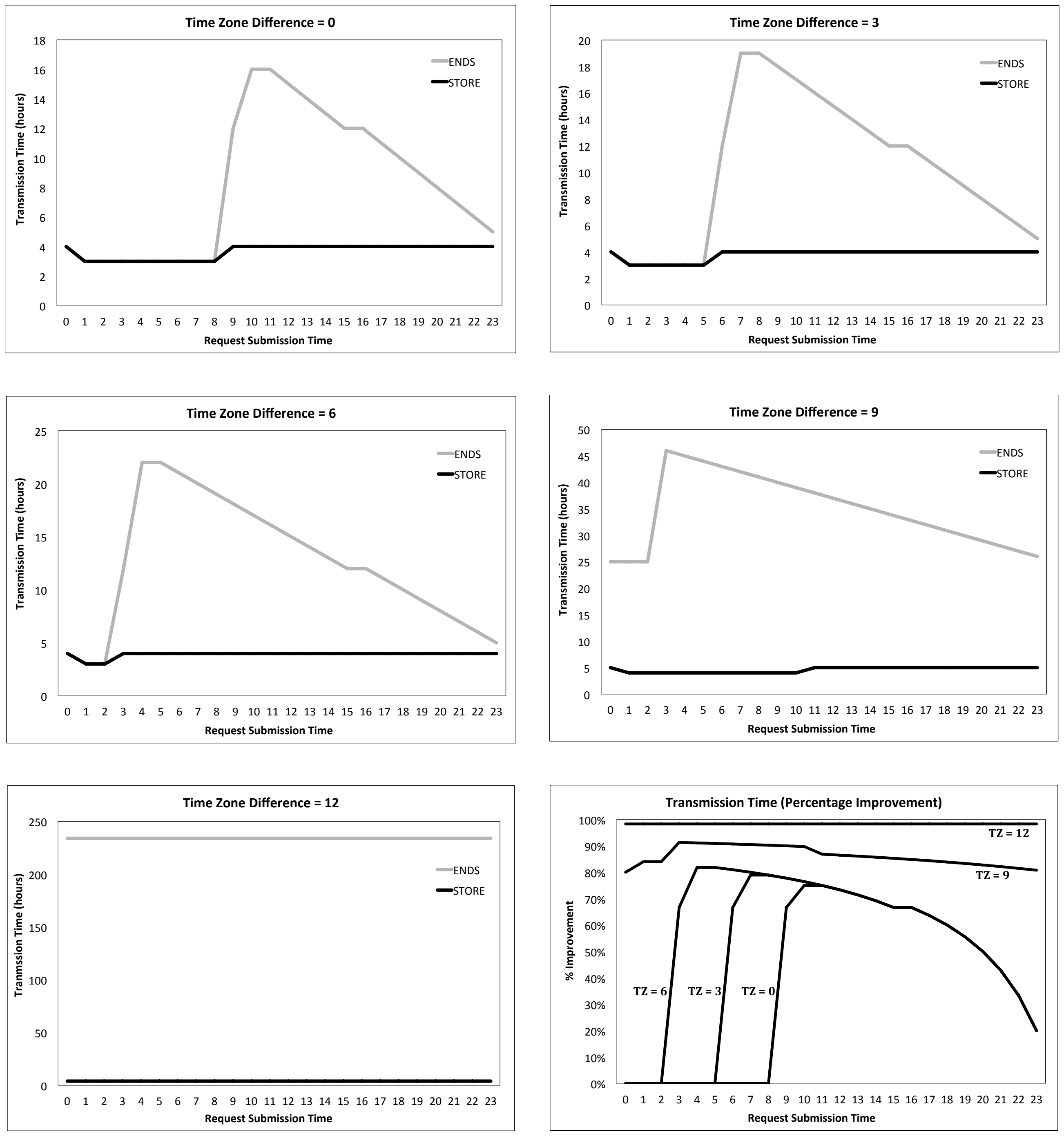
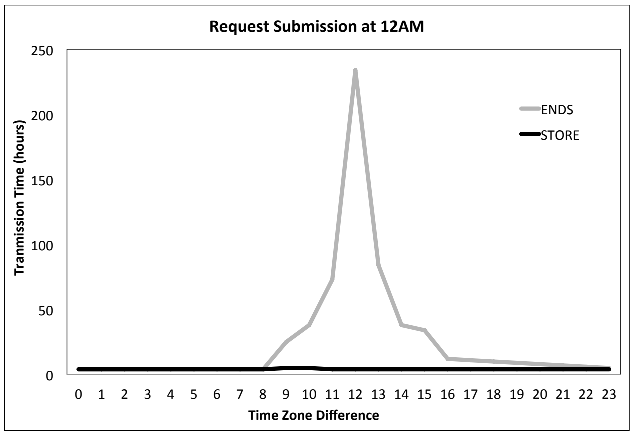


Figure 6: Plotting TT of 1 TB as InitiateTime varies during a 24 hour period. Each graph represents a specific TimeDiff. The last graph plots percentage reduction in TT of store compared to ends.



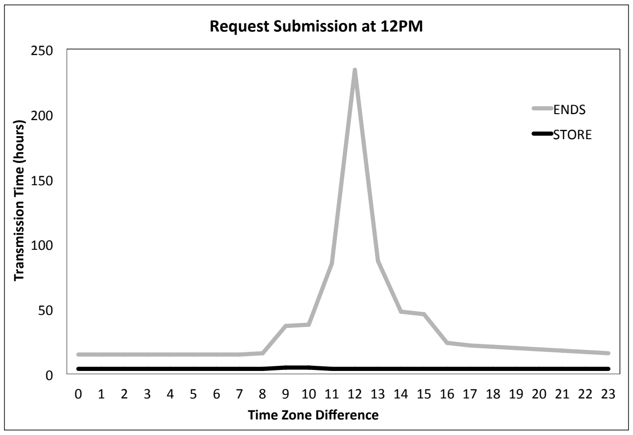
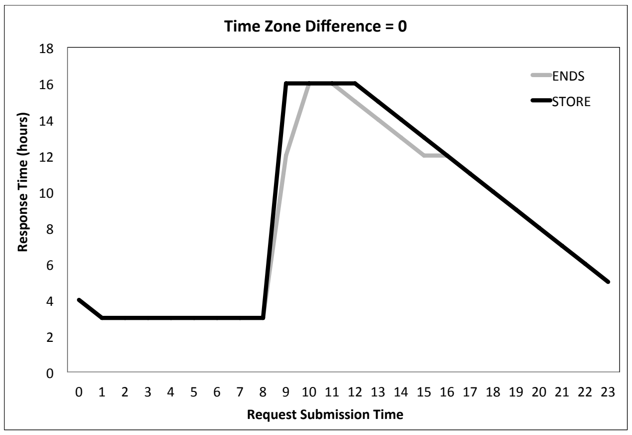


Figure 7: Plotting TT of 1 TB TimeDiff varies from 0 to 23. InitiateTime is fixed at 12AM in graph 1 and 12PM in graph 2.



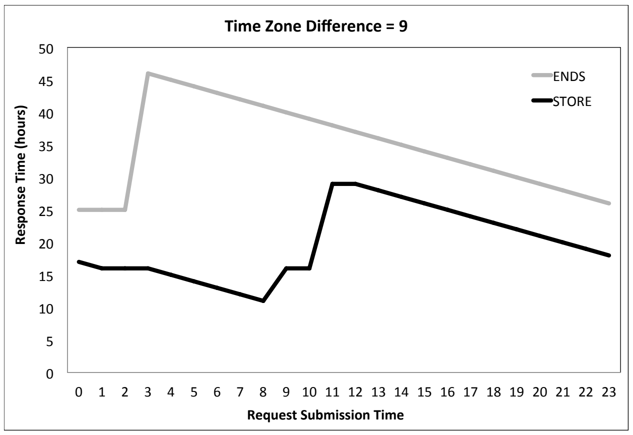
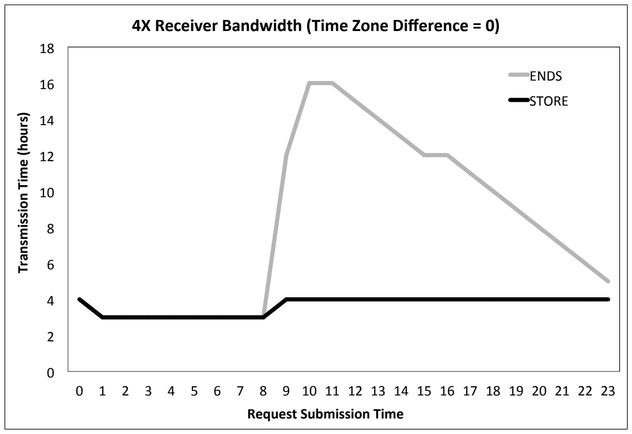


Figure 8: Plotting RT of 1 TB as InitiateTime varies. TimeDiff = 0 in graph 1 and TimeDiff = 9 in graph 2.



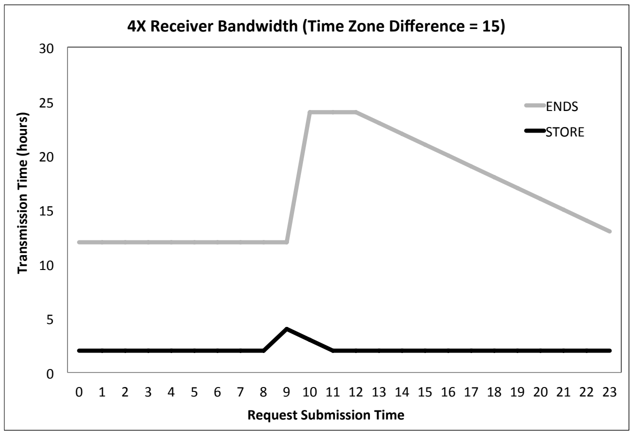


Figure 9: Sender 1 Gb/s; receiver 4 Gb/s. Plotting recTT of 1 TB for TimeDiff = 0 and TimeDiff = 15

5. the bounds can be used as a starting point to determine the routing algorithm between sender to receiver. The bulk transmission routing algorithms are complex since bandwidth is a function of time [5]. The current Store based routing algorithms sometimes make several storage hops in the path from sender to receiver [11, 18]. This may be unnecessary and expensive; the bounding technique, by quantifying the impact of the input parameters allows one to determine the storage hops.
6. the bounding technique can be used for bottleneck analysis. Increasing bandwidth is expensive; before leasing/purchasing bandwidth for bulk transmission, the bounding techniques can be used for quick analysis of the impact of the increase.

8 Conclusion

This paper develops performance models for Ends and Store, the two basic transmission approaches that underlie delay tolerant big data transmissions. The bounds can be used as an optimistic baseline against which the performances of bulk transmission routing algorithms can be evaluated. A contribution of the bounding technique is the simplicity of the performance model and subsequent analysis. The simplicity of the model is a reflection of the choice of input parameters. The big data transmission platform is complex: in addition to the massive data sizes that must be transmitted, the platform includes sender/receiver LANs, several transit networks all managed by independent ISPs, a variety of network protocols, an hierarchy of network routing algorithms, global users, workloads and distances that span the globe (implying varying time zones, user habits dictated by region, different workloads, etc.). The parameters that need to be considered are overwhelming. Abstracting the parameters of significance to the performance of big data transmissions is not easy. The essential parameters are the data set size, bottleneck bandwidths at the sender and receiver LANs during each hour, the time at which the user initiates the transfer, and the time zone differences between the sender and receiver. Using this input, a simple computation outputs when the file would complete download at the receiver. The model quantifies the impact of the input parameters.

Big data transmission routing algorithms have multiple storage hops. Our current evaluation assumes that the storage nodes are not the bottleneck (*i.e.,* the storage bandwidth is greater than the network bandwidth). In reality, storage devices may be the slowest part of a system. As future work, we plan to evaluate the impact that multiple storage hops, proposed in recent bulk transmission routing algorithms, have on the performance of big data transmissions.

References

[1] BLUM, A. *Tubes: A Journey to the Center of the Internet*. Ecco, 2012.

[2] BRESNAHAN, J., AND ET AL. Globus gridftp: What's new in 2007. In *GridNets* (October 2007).

[3] BROSH, E., BASET, S. A., RUBENSTEIN, D., AND SCHULZRINNE, H. The delay-friendliness of tcp. In *2008 ACM SIGMETRICS* (2008), ACM, pp. 49–60.

[4] CERN. Lhc physics data taking gets underway at new record collision energy of 8tev. *http://press.web.cern.ch* (2012).

[5] CHHABRA, P., AND ET AL. Algorithms for constrained bulk-transfer of delay-tolerant data. In *ICC* (2010).

[6] GOLDENBERG, D. K., AND ET AL. Optimizing cost and performance for multihoming. In *SIGCOMM* (2004).

[7] GROSSGLAUSER, M., AND TSE, D. N. C. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Trans. Netw.* (2002), 477–486.

[8] GYARMATI, L., SIRIVIANOS, M., AND LAOUTARIS, N. Sharing the cost of backbone networks: Simplicity vs. precision. In *INFOCOM Workshops* (2012), pp. 171–176.

[9] JAIN, S., FALL, K., AND PATRA, R. Routing in a delay tolerant network. *SIGCOMM Comput. Commun. Rev. 34*, 4 (Aug. 2004), 145–158.

[10] LAKHINA, A., AND ET AL. Structural analysis of network traffic flows. In *SIGMETRICS* (2004).

[11] LAOUTARIS, N., SIRIVIANOS, M., YANG, X., AND RODRIGUEZ, P. Inter-datacenter bulk transfers with netstitcher. In *Proceedings of the ACM SIGCOMM 2011*.

[12] LAOUTARIS, N., SMARAGDAKIS, G., RODRIGUEZ, P., AND SUNDARAM, R. Delay tolerant bulk data transfers on the internet. In *SIGMETRICS* (2009), pp. 229–238.

[13] LIMONCELLI, T. A. Openflow: A radical new idea in networking. *Queue 10*, 6 (June 2012), 40:40–40:46.

[14] LUCIO, G. F., AND ET AL. Opnet modeler and ns-2. In *ICOSMO* (2003), pp. 700–707.

[15] QI, J., AND ET AL. Analyzing bittorrent traffic across large network. In *Cyberworlds* (2008).

[16] RAMAKRISHNAN, L., AND ET AL. On-demand overlay networks for large scientific data transfers. In *CCGrid* (2010).

[17] SHERWOOD, R., BRAUD, R., AND BHATTACHARJEE, B. Slurpie: a cooperative bulk data transfer protocol. In *INFOCOM* (2004), pp. 941 – 951 vol.2.

[18] SHI, C., AMMAR, M. H., AND ZEGURA, E. W. idtt: Delay tolerant data transfer for p2p file sharing systems. In *GLOBECOM'11* (2011), pp. 1–5.

[19] STANOJEVIC, R., AND ET AL. On economic heavy hitters: shapley value analysis of 95th-percentile pricing. In *IMC* (2010).

[20] VENKATARAMANI, A., KOKKU, R., AND DAHLIN, M. Tcp nice: A mechanism for background transfers. In *OSDI'02* (2002), pp. –1–1.