Data Traffic and Bandwidth Analysis Hybrid model to combine Peer to Peer & Client Server models

Venkata K. Kishore. Terli, Abhinav Risal, Sundeep B. Chavali, Kailashnathan Thirupathur V. R., Anvesh K. Pedakotla and Omar Abuzaghleh
Department of Computer Science & Engineering
University Of Bridgeport
Bridgeport, USA

Abstract—This paper addresses current data and bandwidth problems by re-organizing few limitations of server-client and peer to peer communication model so that a hybrid model would solve issues related to both of them. The current infrastructure has some issues in catching up with the vastly increasing speeds and requirements of clients. Following these trends, the existing infrastructure needs to be completely revamped in order to successfully meet the current requirements and also the requirements of the near future. Here arises a need to meet the current flow of demands while also buying time in order to get the new infrastructure in place. In order for that to happen, there exist some factors that can be manipulated so as to optimize the current system to meet the demands and provide a time buffer. Here these factors are researched upon and conclusions are drawn from this research so as to get the ways to manipulate and use these factors/resources in a correct manner for managing data traffic.

Keywords—bandwidth, p2p, s2c, data limit, multiplexing, dynamic, hybrid p2p communication

1 INTRODUCTION

The Internet has become the backbone for modern day communication, business and multimedia related services. Modern internet has unfolded several groundbreaking capabilities leading to increased content sharing, globalization, and boosting number of mobile internet devices. This increased potential for utilization of the internet and its related services has led to the number of users connected to keep growing on a daily basis. As this number grows so does the amount of data traversing the network, adding further strain on the existing infrastructure. Ever since its conception, the basic infrastructure of the internet has remained the same. This stagnation in infrastructure development has resulted in some serious drawbacks such as reduced speed and quality of connection, service provider uplink limitations, and uneven distribution of services to consumers, among others. While an increase in overall bandwidth would be an ideal solution, it involves a serious cost factor in order to overhaul the current network set-up. Even then it would not be an effective usage of the available resources. This paper expounds on several methods such as dynamic speed distribution and dynamic band width allocation which may be implemented in order to address some of the issues mentioned above and in the long run provide a means to increase effective utilization of the current system.

2 PREVIOUS WORKS

The main problem here is that the demands for data are increasing and also the usage of data and data dependency of humans is increasing. This is affecting the day to day functioning. So when it affects the day to day functioning it means that it is becoming a crucial part of our lives. So to meet the demands the data needs to be delivered according to the needs. So there were a lot of proposed solutions.[1, 2]

A lot of the technology currently being developed like the advanced voice recognition and control systems have extremely high physical requirements for back end functionality. The voice guided systems which are increasing in popularity and the similar technologies in development could one day replace our current devices and may even take up a large portion of the market in the very near future. As we draw closer toward realizing such innovation, we are reminded harshly of the current infrastructures inability to cope with the demands this would present. A voice guided search engine would tower over traditional text based search engines like google in terms of physical data servers and computational power. This would result in exponential increase in the cost factor of implementing such systems as well as the energy required to keep the system running.

First one which is in the problem scope of this statement is to change the infrastructure that has been laid down in all these years[3]. So all the things existing need to be changed from scratch and needs to be modified with new structures. For examples co axial cables need to be replaced with optical

fiber cable and so on. This solution is actually good as this takes care of not only the current scenario but also augurs well for the near future when the demands increase. But here this solution requires changes to be made from the core level. This changes require a lot of money and time be spent for this changes and people are reluctant to spend either. As getting out the new structures takes a lot of money this needs some time so as to accumulate the required resources. And for the acquired resources to be correctly placed and managed effectively, it would take time. Because the changes are not done in one place, but each and everywhere. So this cannot be done in an instant which in turn means a lot of time would be required to make it possible. These effects the existing functions. So this paper gives a short term solution so that the changes and modifications can be done without affecting functionality.

The Other solution that was implemented was to increase the server capacity[4]. What happened here was that the clients systems are becoming more and more efficient and are generating more traffic. Then the servers up gradation was the solution as it was seen that due to the server being slow, the speed was reduces and also the effectiveness was also affected. So they upgraded the servers and client request handlers and data providers. So here everyone is looking at the endpoints in the current chain. This means that there was something that was being overlooked. The thing that was being overlooked was the fact that the data was being sent and also being received, but how was it being sent and received? It was through the transmission medium. So here no matter how much we speed up the clients systems or how much we speed up the Servers, they transmit data through these communication channels. And these were not cared for and also was thought to be able to transmit whatever data at whatever speeds. But this was the main part. Even though a lot of data is sent from the systems, it needs to travel through the tunnel called transmission medium. This medium was small that it cannot let all the data flow through at once. So this approach ended up creating more trouble rather than solving the problem.

So here we are taking this into account as both the client and server are being upgraded, we need to upgrade these and we are finding ways to upgrade it. But then up grading it is viable in the short term. This problem needs to be solved for both short term and long term. So as we saw that the long term solution which is good is to completely revamp the infrastructure, we are trying to find ways to optimally utilize the existing resources so as to be able to solve the problem for a short term. While looking at and analyzing the factors that determine or control the data traffic.

After initial research we found the factors that are the key to solving this issue. As mentioned in the Introduction, these factors will be dissected individually and also at the same time combined to search for the solution. The paper will also discuss possible methods to implement the solution.

3 HYBRID PEER TO PEER COMMUNICATION MODEL

Using peer to peer, we all know higher file transfer rates can be achieved, but can this help in addressing data traffic? Can it help improve our streaming of videos any further? What could we do to enhance our streaming speeds and net browsing experience? In this paper we give advantages of using peer to peer system not just as an application specific to one unique purpose but to improve internet experience.[5]

We propose to establish a new network model which enables peer to peer file transfer easily and conveniently to customers by enabling them to use their true internet service potential. This browser should be able to notify server about the file being requested. The server should assign this request to all other users seeking same content. Now the file is divided into pieces and then shared to each client requesting it. However all clients will then also interact with each other uploading their piece of file received as a natural gesture when both upload and download is justified. When content with more demand is requested on a client server model, all servers would have to increase their speed or boost their bandwidth. However, in this an ideal scenario if current requesters are unlimited then we don't have to send file anymore as all cleints could share their part of file and still be able to have very fast file transfer rates. Integrating security and monitoring copyrights will be a challenge for us, this paper will analyze that to some extent and help in achieving more data rates with existing infrastructure. Further, we will analyze all the results in detail.

In a regular peer to peer connection, each peer has to download from others and share its part to support the peer grid or peer web. In earlier generations of internet when network protocols are not so much evolved, infrastructure and basic connectivity was a problematic. Peer to peer, were promising as they helped in getting data faster and reduced burden on servers. As the infrastructure and governing protocols have evolved all data packets and connections can be controlled by the server effectively. [6]
Peer to peer systems can thus support hundreds and thousands of systems with same capacity as of a server client model.

Proof:

Let each peer be "p" and group of such "n" are controlled by a server whose capacity of upload be "U" and download be "D" group elements each peer and server together will have a group capacity of upload (GU) and download (GD).

GU is the sum of each individual group upload

$$GU = (\sum Pi + Ps) \text{ upload}$$

$$= \sum Ui + Us \quad (1)$$

For group downstream

$$GD = (\sum Pn + Ps) \text{ download}$$

$$= \sum di + ds \quad (2)$$

From (1), (2) where each peer agrees to contribute to the network but in reality we have fewer uploads than downloads. The upload speed of server is also affected by connections handled. Each peer connection is valid for only certain time (Tq) in the web, this is called 'Time Quantum'. Once it is over peer moves to an idle state. The server has to send an acknowledgement (ACK).

For entire web, $\sum Tk$, the acknowledgement takes this time to show the web but they are characteristic features on which protocols rely. Let's analyze further to see the final result.

Effective upload and download:

The peer to peer group could be beneficial to each individual system but they as a group would reduce burden on server. This reduced burden would be the effective gain to its server. Gain function overall can help server by

Assumptions:

4. Downstream speed = upstream speed = d

   Number of connections = n

   Server capacity = M

Resultant speed or server capacity after or on serving of all these,

$$R = M - (n)*(d)$$

For peer to peer Rp we have all small clients or peers uploading with capacity of n(nu)

$Mp = M + n(d)$

$Rp = Mp - (n)(d)$

But ideally if a file has 2 pieces A, B. A is shared with half of the keys and B with the other half. So for single file transfer from server will yield into only half of server capacity. Ideally, server is half occupied on realistic basis. According to [8], the Cumulative Distribution Function (CDF) of a peer's download bandwidth can be modelled as a linear function, which means that the PDF of a peer's download bandwidth can be modelled as a uniform distribution, i.e., ri's are uniform.

For the given peer web if the lowest connection speed of a peer be dl and its upload is the lowest to be dl. These dl and ul are always less than the group upload and download speeds. This peer can't receive more than its download rate or send higher than its upload rate. [7]

When we don't have any idealistic conditions but still near idealistic networks can help us in analyzing this. The following are the parameters for the network.

Let probability be "p"

R = download rate for each peer

N= no. of peers

N peers and download rate = r*(n)

This is for ideal case

$i=0\sum N$ ri
= when user uses as peer to download / upload

Since P(r*n)*G>>Negligible

Even here group will keep servers busy in one cycle and is free in the other cycle

So in avg. it is only half the capacity occupied.

Effective capacity

C|=C/2.
Using this model we have effectively gained 50% of server capacity. The capacity Cg is the gain when compared with a full load server client model Cs is as follows.

Cg= (C-C/2)/100= C/200=50%(C)

4 DYNAMIC BANDWIDTH ALLOCATION

The bandwidth can be imagined like a freeway. The higher bandwidth, the more lanes you have for data to drive through. Usually a higher bandwidth means high speed. But even if we have a reasonable bandwidth we still don't get the required amount promised. Even with a high connection speed multiple users will have a difficult time because of delay in the transmission of data from the internet. The bandwidth even if promising high content volume in adequate speed is always not able to provide as promised. [8]

Dynamic bandwidth allocation algorithms have been proposed where for the allocated bandwidth will be divided

and according to priorities. These priorities are divided upon the Quality of service which can increase the effectiveness of bandwidth by prioritizing among and delivering time sensitive packets first.

Three priority[9, 10] levels have been used for classification. First, the high priority service supports applications that require bounded end-to-end delay and jitter specifications. For example, it is used for voice and constant bit rate video services with low jitter requirements. Second, the medium priority service implements a traffic class for applications that are not delay sensitive but that require bandwidth guarantees. It is used by variable bit rate services that are non-real time service. Lastly, the low priority service is provided to implement a best effort traffic class. It is not sensitive to end-to-end delay or jitter. Using the concepts of the priority levels we propose a method of dividing the allocating bandwidth among users so that each will get their required optimal connection without hampering other connections and maintaining the integrity of the network network. The idea is to distribute bandwidth fairly between users so each one will be having a reliable connection. [11]

The internet uses many services which consume bandwidth at different rates. These services may be differentiated and categorized based upon the bandwidth requirements. Service may get slow or delay if the proper bandwidth is not available. Bandwidth is unlimited in a point to point network but it is limited in a shared bandwidth limited leased line system.

Below we propose four different kinds of services that consume bandwidth in descending order. Based upon this we apply an algorithm to give the required bandwidth to each without slowing down any service. The four services are described below:

1. Media

Media consumption was once of very low priority but now it's driving all intent with various forms. Websites are loaded with images or media content. They need high bandwidth and superfast speeds of data to be consumed. This service gets highest priority in our system. Thus making it a media biased system where the priority of media is high.

2. E-commerce

Ecommerce is the next preference, a trade works on inter connected networks and heavy volumes of data from stock markets, IT firms, government sector are sent and received every other day. These take second priority.

3. Static data

Static data, many content providers like Wikipedia, google or yahoo, have static data which is presented to any user without time dependency. Most of the data is static and can be accepted at normal rates of speed. These can be kept in a low speed line of internet without bombarding data. This data can be cached in a few search engines and be served as and when required. In doing we can stop burden on back bone networks of internet.

4. Internet of every other thing (IOEOT)

Moving on to a smarter planet every day, we can use this service to be dedicated, such as for data carriers such as sensors. All the latest sensors or human, planet, agriculture, weather, pollution data etc. can be sent over this lane which is of low speed but highly secure. As data from these sensors can be accommodated in very low memory, this data can be sent as bulk and can be managed on the last fast lane.

TBW = Target bandwidth, EBW = Expected bandwidth, RBW = Required bandwidth

$$BWmain = \sum_{i=0}^{4} BWi \quad (1)$$

$$=BW1+BW2+BW3+BW4$$

$$TBW > EBW$$

$$EBW = TBW-RBW \quad (2)$$

Condition 1: EBW >= 0 (Stable Condition)

Condition 2: EBW <=0 (Unstable Condition)

For condition 2 we have another 2 sub cases

A) Take the bandwidth from the lowest priority in reverse order

B) 2 is full then people from 3 and 4 at speeds of 1 and 2 will might be interested in sharing their speed and this shared speed will further be used to increase the effective bandwidth of 1 and 2.

Stable and unstable. IN priority 1 we again will divide it into 2 lanes. Where one is for live streaming and the rest is for media consumption.

5 MULTIPLEXING RELATED DATA

What is multiplexing? Commonly we can say that multiplexing is combining the data together for faster transmission and also to make a more efficient use of the transmission medium. We would mostly combine the related data together to increase the Data transmission efficiency. Here we are reducing the transmission costs over a medium by multiplexing. So the usage of this process dates back to 1800's when it was used for telegraphy. But this is more

widely used in many telecommunication applications now-a-days.

In this we mainly used a Multiplexer for combining the input signals together, be it analog or digital or both and then infuse them to make a combined single signal. And on reaching the destination, the signal is then separated to its component signals by the use of the Demultiplexer. But Multiplexing can be sub divided into more categories such as Time-Division Multiplexing, Frequency Division Multiplexing and also Code Division Multiplexing [12]. To explain these further, we need to look into each of them separately

Time Division Multiplexing (TDM) – This is a form of multiplexing where the lines connected to the multiplexor are combined in a time wise manner. Here the multiplexor is assigning time splices to each of the signals and them transmits them. Also there can be synchronous and asynchronous TDM.

In synchronous TDM, each of the lines is allocated a splice of time regardless of the requirement of the line, this lead to lesser throughput because the line is being allocated some time even though it is idle.

While in Asynchronous, the time is only allocated to the next sending device in line, if the current sending device is not sending any information. But this requires additional processing and may lead to delays.

There is one more type of TDM that is Statistical Time Division Multiplexing (STDM). In STDM, the Sending devices are assigned time slots neither randomly nor directly to the next device in line which is ready to send, here the time splices are assigned based on the Statistical information of the sending device. The main statistics used in STDM are: each input device's peak data rates (in kbps, or kilobytes per second), and each device's duty factors (which is the percentage of time the device typically spends either transmitting or receiving data). [13] Here we can send multiple packed being sent by the same peer in p2p model and then multiplex all of these into one big packet of data. This also saves the space as many different headers are not needed.

Frequency Division Multiplexing (FDM) - This is a type of multiplexing where data being sent are each put into different frequencies and then all this data together are transmitted. Then at the demux, the data is differentiate based on the frequency and then sent to the corresponding device.

Code Division Multiplexing (CDM) – In code division Multiplexing, the Input signal is modulated along with other signal which would be orthogonal to this signal. Which means that both the signals are not interfering with the other's data. Then this signal has a code attached in order to provide checksum of the transmitted signal.[14]

With the current techniques in multiplexing, the speeds currently are almost reaching a bottleneck. This is being overcome with the efforts in modifying the current schemes so as to accommodate for higher speeds. The need to find new ways to improve the speed until the underlying network could be modified so as to handle such speeds effortlessly. Actually many are coming up with their own schemes to overcome the hurdles of the data traffic. Each discipline or person is turning to their own needs and solving it so as to reach a solution. For example, the gaming industry has a lot of online games in which loads of data need to be transferred at real time speeds. In this regard using general methods with the given infrastructure makes it is almost impossible or not cost effective. There are hybrid models being developed to handle these situation in that specific industry. Below is a diagram of such a model. This illustrates the combined server architecture and the integration of the game clients with the two types of architectural model. [8]

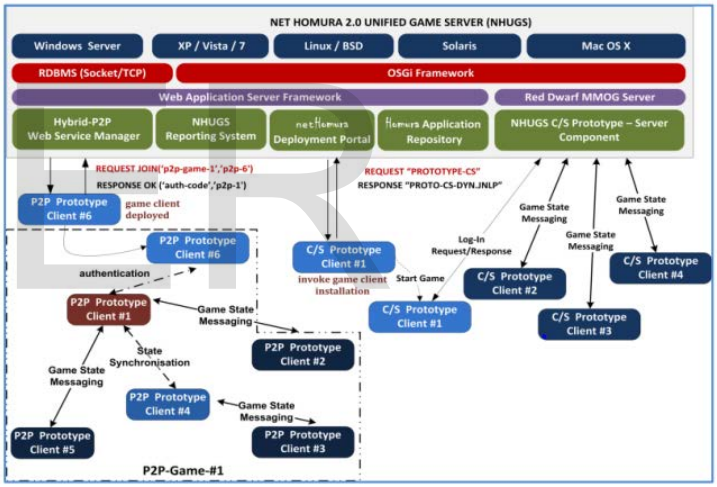

Fig. 1. A Hybrid Model

Also for ATM networks, there is research being done, which will be used by many people across the world to enable access and modify bank accounts. There are strategies being developed to minimize the load and maximize the throughput. Here, a formula was developed for allocation of the time or bandwidth or frequency as such. The parameters presented to the sentinel are:

1. 'm': a representation of the average cell rate as used in the allocation formula

2. 'sd': a representation of the deviation from the mean and used in the allocation formula.

3. 'Tc': indicating the time constant of the service all of the incoming data cells will be checked on its effect on the limits.

If the cell makes the expected values of the connection surpass the limit values, the cell will be taken out of the incoming stream. Otherwise the measured values will be adapted and the cell will be passed to the network. [15]

$$\sum_{i=1}^{N} (mi + a * sdi) + B * \sqrt{\sum_{i=1}^{N} sdi^2} <= \text{max. capacity}$$

$$\text{with } B = b * c$$

Using such algorithms and then modifying them so as to work for general networking on the whole, the current infrastructure can accommodate the speed for the near future. But we need a specific multiplexing scheme that can be applicable to most of all the networks and patterns. As multiplexing should be fast, energy efficient and should not affect the network throughput, SAR A/D converters are widely used for multiplexing, Using these General schematic for multichannel systems would look like this.

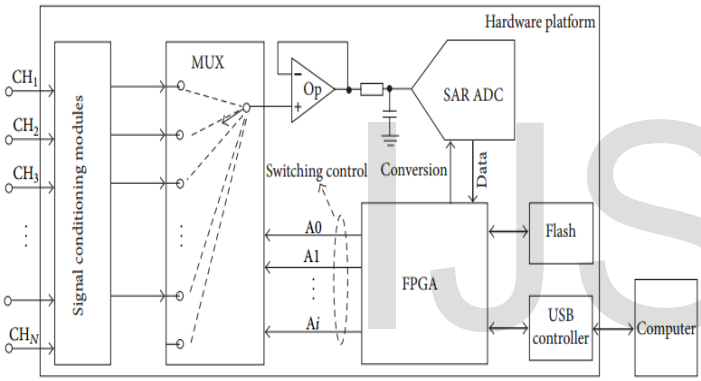

Fig. 2. Encoding and Decoding of signal.

So here we make up the multiplexing scheme by sampling the frame format and then using the FPGA-Based Multiplexing Scheme Lookup Table. Compared with the register and related logic control design method, the FPGA-based multiplexing scheme lookup table method greatly reduces the internal resources and improves software quality and stability. [7]

By using these schemas and combining the schemas of other disciplines efficiently the usage of the network can be increased and the load can be reduced. The increase in throughput will be easily achieved by combining the multiplexing methods along with the existing infrastructure. Also we can look into combining Statistical Time division and frequency division multiplexing. By efficiently combing STDM and FDM, we can Allocated Time splices in certain frequencies based on the network statistics for each sender thereby multiplexing the signal with advanced catering capabilities.[16]

6 DYNAMIC SPEED DISTRIBUTION

The speed of a network connection is a highly coveted resource. It has unanimously become the deciding factor for selecting between service providers. Since the dawn of the commercial internet around thirty five years ago, it has grown in leaps and bounds. A majority of the credit for this feat goes to the technologies and hardware behind the scenes in a properly functioning network structure, from the very basic elements like ethernet cables and ports to gateway devices and access points (and the protocols governing them). The development and implementation of these parts and the continuous efforts to furthur improve on their functionality has allowed for the realization of the high speed networks of today. With this development and growth in network speeds there has been a subsequent increase in network data flow issues. These issues breakdown the optimal functioning of the network and result in low speeds at end users. In this section of the paper, we propose to show better effieciency and higher throughput by modifying some traffic shaping as well as congestion control techniques and comparing them with previous works.

When speaking about network speeds and the subsequent upload and download limits, we must first understand how the limits are put in place and how the data traffic flow is controlled. Although the term speed is used quite freely, and is often confused with bandwidth, it can be simply defined as the maximum (or optimal in most cases) flow rate of date through the connection. In other words it literally means data rate (bits per second).

For the purpose of this paper, assume that the flow of data is in the form of packets. In a network with multiple nodes in full duplex, there would be packets waiting in an outgoing queue for each node transmitting data, as well as packets waiting in a sort of incoming queue at the receiving node waiting to be processed. Assuming this network state to be true, there are several scenarios which could result in causing the network to crash. For an example consider a fully connected network of five nodes as shown below.

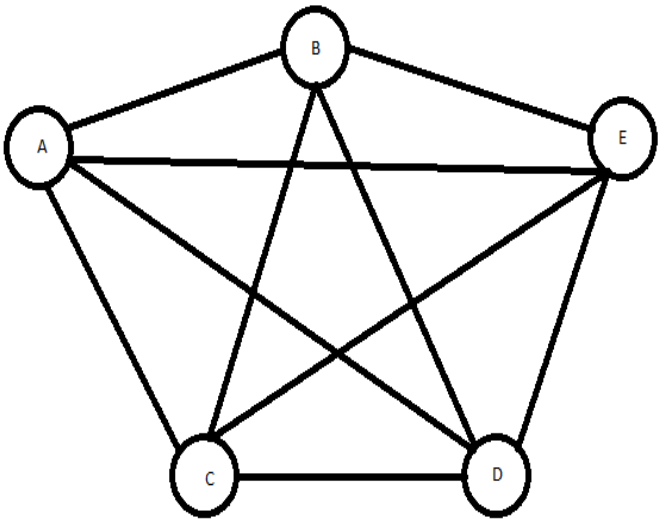

Fig. 3. A fully connected network consisting of 5 nodes.

In an ideal state, all five nodes would be sending and receiving data simultaneously. However in the case of heavy load on the network, there would be a noticeably large propagation delay and packets could even be lost. Consider a transmission from A to C, while there is a dedicated connection between the two nodes, in a heavy traffic or congestion situation node C could have a full incoming packet queue. This could result in the packet from being discarded or a timeout resulting in a retransmission which would put further pressure on the burdened network. Scenarios such as the one explained above result in an inefficient network with a low throughput. In such cases of congestion, with further increase in network load the throughput increases linearly up to a point and then stays constant for all further increases past that point. When this critical load point is reached, average delay per packet and overall network delay increase rapidly.

There are several methods already in place to address these issues of network performance and in order to better handle or avoid such situations. These include:

Backpressure: Where a node suffering from congestion can slow down or stop flow of packets (incoming) from other nodes. This is achieved by applying flow restriction in a backward propagating manner to the sources. This method can also be used to stop traffic selectively while allowing flow on a few particular connections.

Choke Packet: This is a very simple mechanism to control the congestion on a network. The choke packet is a control packet which is sent by the congested node to the source informing it of the congestion. The choke packet is sent whenever the receiving node is forced to drop or discard a packet due to congestion. The sender receives this packet and resends the dropped transmission but holds off on other/new packets until it stops receiving choke packets.

Implicit and Explicit Congestion Handling: These are internal mechanisms used to forewarn other parts of the network regarding a possible congestion.

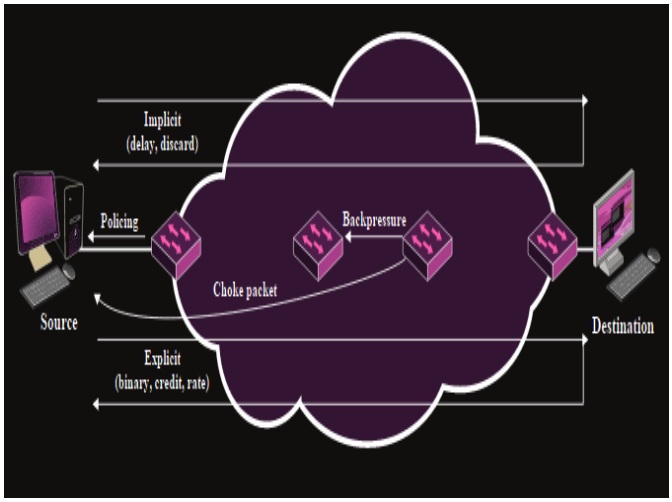

Fig. 4. Implicit and Explicit Congestion Handling

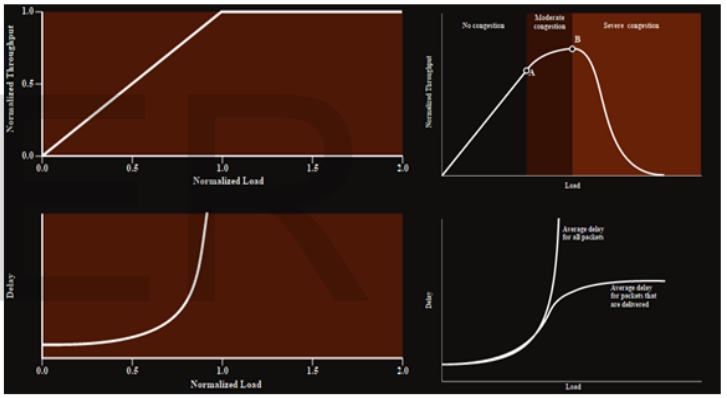

Fig. 5. Load vs Delay

- DCCP: The Datagram Congestion Control Protocol is an alternate transport protocol for applications which generally use UDP. The Internet DCCP Working Group has defined two congestion control mechanisms from the DCCP. They have unique congestion control IDs (CCID)
    - TCP-like Congestion Control (CCID 2) is used with applications that require maximum utilization of the available bandwidth but do not require a constant or high data rate.
    - TCP-friendly rate control (CCID 3) is used with applications which require a constant data rate throughout the connection. Eg: streaming media.
- Traffic Shaping Algorithms: There are two algorithms commonly used to regulate the average rate of data flow between sender and carrier at the time of initial connection set up. These algorithms essentially police the various traffic generating sources (applications) and provide a concise limit on the load that can be imposed

by a flow. They are the Leaky Bucket Algorithm [17] and the Token Bucket Algorithm [15, 17, 18].

- The Leaky Bucket Algorithm: This algorithm is used to control the data rate in a network and it is implemented in a queue structure. The purpose of this algorithm is to maintain a predetermined output rate and thereby oversee uniform distribution of traffic on the network. The algorithm follows three basic steps; for each passing unit of time, a certain number of conforming tokens are added to the bucket, for the same unit of time the total capacity of the bucket is incremented by the number of tokens added, finally a certain number of tokens are released (leak out) from the bucket per time unit.

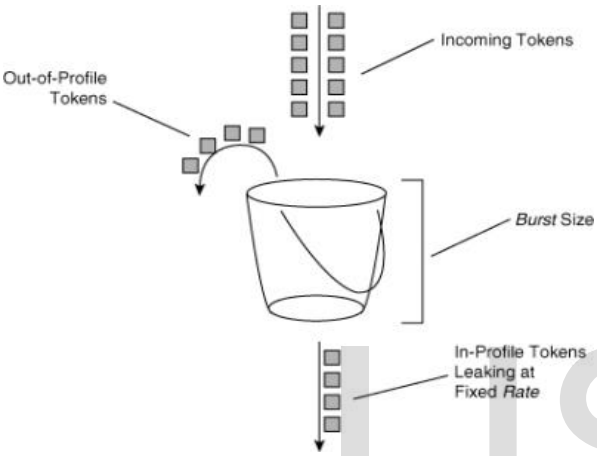

Fig. 6. Leaky Bucket

- The Token Bucket Algorithm: This algorithm is very similar to the leaky bucket algorithm with the only difference being in the limiting of the token discharge rate. Here too there exists a virtual bucket capable of holding a maximum number of tokens n. These tokens are released one at a time per unit time as opposed to the continuously released tokens in the leaky bucket. The token thus released is then provided to the packet at the head of the queue which then proceeds into the network.

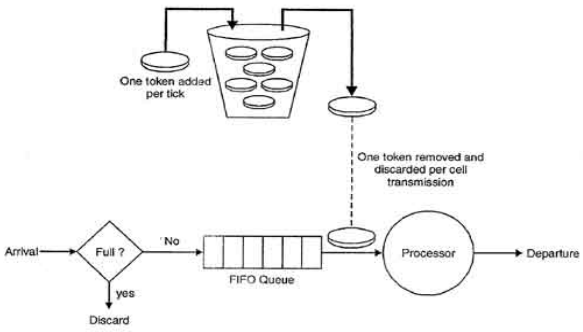

Fig. 7. Token Bucket

In order to better understand how these algorithms actually help maintain a data flow rate, visualize a ticket attendant's stall at a carnival ride's entry point. People would not be able to enter the ride until they have received a ticket from the attendant. Here the attendant would be representative of the algorithm, people would represent packets and they would be waiting to enter the ride which would in turn represent the network. In the case of leaky bucket algorithm, the attendant could continuously keep handing out tickets at a fixed rate and the people would be able enter the ride in a first in first out (queue) manner. This seems like a very simple method to control the traffic flow, but its expected functionality is hinged on people/packets exiting close to the rate at which they entered, as were the rates different it could result in traffic build up and congestion in the network. In the Token bucket algorithm, the flow is slightly more restricted, as the attendant would only hand out one ticket for a certain time interval, thereby allowing more time to free up the ride. In this scenario there would be more time for a packet to complete the traversal of the network and for the receiver to free up more space on the incoming queue so that the packet does not have to be dropped. While the above examples seem to be pointing to ideal condition functionality of the algorithms, they would only apply in the case of fully connected networks of point to point dedicated links. A regular network would have multiple senders and receivers at the same time, and also all nodes would probably not be fully connected. Considering a larger network, there would be more overall network traffic and on more than one occasion multiple senders could be sending packets to the same destination, at the same time. This could still result in overloading the incoming queue of the destination and thereby resulting in packets to be dropped. To avoid these issues altogether the algorithms must be modified, other methods applied to check the overall data being input into the network and a system must be implemented to observe the network and report on possible congestions.

Over the years several papers have taken the Token Bucket algorithm and tried to modify it, add to it, or apply it differently to achieve closer to ideal functionality. Some of the notable works using the Token Bucket Algorithm are summarized below.

One of the most common papers related to this algorithm is the Hierarchical Token bucket [17, 19]which uses the token bucket algorithm as an alternative to the rat scheduler to allocate bandwidth to classes in the papers' alternative implementation of the Class Based Queueing (CBQ). A different paper observed and recorded the performance of the Hierarchical Token Bucket in IEEE 802.11 [18] to enhance the quality of service. They concluded that overall HTB allowed for a better quality of service as well as a higher throughput. An alternate approach was the Token Bucket data in an FBM [17, 20] to analyze the burst densities of different applications and thereby modify the Token Bucket parameters. This

resulted in higher Hurst parameter traffic having a faster decreasing burstiness function.

We propose an alternate version of the token bucket algorithm which would function as part of the hybrid network model to better control data flow rate and achieve close to ideal operational functionality. In this modification of the token bucket algorithm the general structure of the algorithm would be the same. There is still a bucket with a maximum capacity of 'n' tokens, feeding out a single token for every fixed time interval 't' and tokens (m) are added to the bucket at the same fixed interval provided the bucket is not at full capacity (l).

No. of token in bucket (l) + No. of tokens added (m) < Max No. of tokens that bucket can hold.

L + m < n

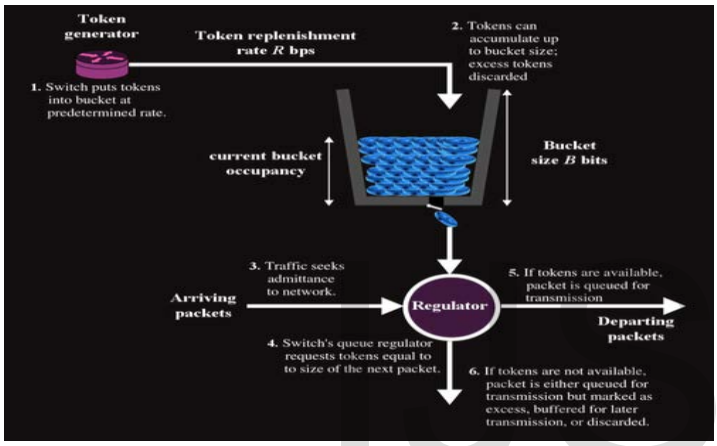

Fig. 8. Working of Token Bucket

In the original algorithm each token released at time interval 't' is assigned as a ticket to the packet first in the queue to enter the network. There is generally no distinction between packets as to their size, origin (application source), or priority. Packets originating from any of the various applications will end up in the same queue together. The alternate version of this algorithm would have multiple queues and they would be categorized by priority. As shown in the figure below, there would be three different priority levels. Priority level 1 would be reserved for network critical and time critical functionality. Priority level 2 would be a flexible level assigned upon request from an application or from the central server. Priority level 3 would service all the generic range of packets.

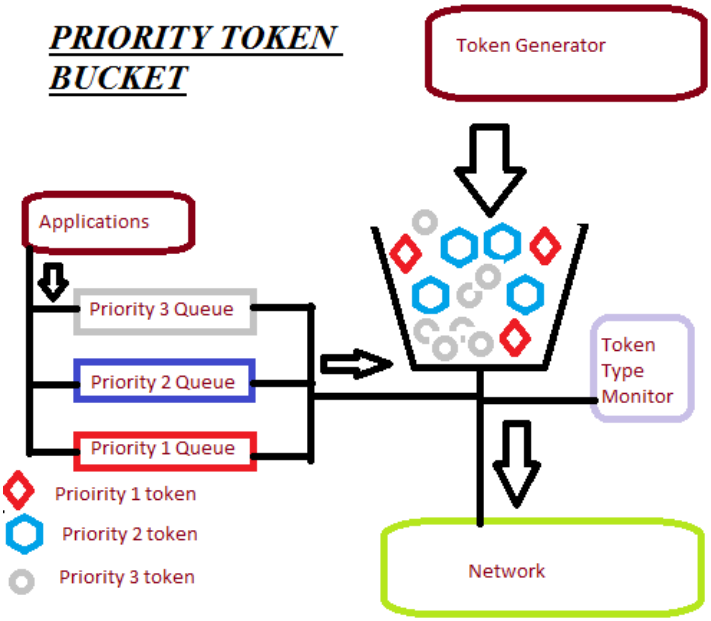

Fig. 9. Priority Token Bucket

The assigning of a packet or a class of packets (for a whole application) to a higher priority is carried out on a case by case basis upon request by either the user, or the server. By default all non-critical functions are given priority 3 and priority 2 is kept vacant. The hybrid network model explained earlier in the paper is a combination of the client server model as well as the peer to peer architecture. The purpose of this new model is to enable faster data sharing rates and reduced server load. The priority based token bucket compliments this idea by having different priority among tokens as well as among packets. When a new packet comes in from an application it is sorted into one of the three queues to await a corresponding token of same priority level. When the packet grabs the appropriate token it moves onto the network. The tokens leave the bucket based on required priority type of the next packet. Priority 1 queue is given first shot to leave based on availability of tokens. However priority 1 has fewer tokens than priority 2 and priority 2 has fewer tokens than priority 3. This is to ensure the bulk of the network traffic based out of the user applications does not get stagnated by priority based system.

The tokens generator maintains a count of total tokens available in the bucket and based on the size of the bucket generates token until full. The token type monitor keeps track of the type/level of token released from the bucket and informs the token generator. It is the responsibility of the token generator to maintain the balance of ratios between the number of tokens of each priority level. This ensures there are always the same number of tokens of each type, which means the parameters that define the token bucket functionality are maintained. These parameters may be changed to allow for

different ratios but once saved to memory they must maintain until next change.

When a user requests for a certain program or process to be given higher priority, all the packets under that class are temporarily elevated to priority 2. This gives the program access to a more dedicated network as priority 2 is temporarily monopolized by it and thus results in faster functionality. Once the program terminates, the end system shuts down, or the node is disconnected from the network, the priority is reset to level 3. This implementation when used alongside the hybrid network model allows the server to assign priority for particular outgoing transmissions. If one of the nodes on the hybrid network requests a large file transfer, the server would keep track of prior requests for the file, and dynamically split the file into parts and assign a part to each node having the file to transmit to the requesting node. The requesting node ends up receiving multiple parts of the file at the same time from different users. From the downloader perspective it is almost as if the file downloaded immediately. This method with the token bucket algorithm would allow the server to send transmission requests to all nodes that have a copy of the file with priority elevation request for the process to ensure quick delivery. Overall such an implementation would be able to achieve better efficiency over the network (due to its utilizing both upload and download limits to maximum extent) and could also achieve close to optimal and congestion free network operation.

7 CONCLUSION

This work uses a compilation of alternate network structures, varied dynamic bandwidth allocation algorithms, concepts of multiplexing, and congestion control techniques to develop a new and unique network model. This hybrid network model is comprised of a merger of two common network structures with an improved bandwidth allocation algorithm and redesigned multiplexing and congestion control techniques to bring about a system more efficient, economic, and optimal functionality in a small to medium scale network. The proposed model has the potential to outperform traditional systems and could help to provide a temporary solution to most speed and bandwidth issues without resorting a complete network overhaul in the near future. Future work in this area would focus on implementing the suggested techniques and observing the mathematical variations between both systems.

8 REFERENCES

[1] Choong-Soo, L. The revolution of StarCraft network traffic. in Network and Systems Support for Games (NetGames), 2012 11th Annual Workshop on. 2012.

[2] Amin, R., Analyzing performance of ad hoc network mobility models in a peer-to-peer network application over mobile ad hoc network. in Electronics and Information Engineering (ICEIE), 2010 International Conference On. 2010.

[3] Ortiz, S., Internet Researchers Look to Wipe the Slate Clean. Computer, 2008. 41(1): p. 12-16.

[4] Jie, H., L. Yaping, and G. Zhenghu. Support mobility for future Internet. in Telecommunications Network Strategy and Planning Symposium (NETWORKS), 2010 14th International. 2010.

[5] Koo, S.G.M., C.S.G. Lee, and K. Kannan. A genetic-algorithm-based neighbor-selection strategy for hybrid peer-to-peer networks. in Computer Communications and Networks, 2004. ICCCN 2004. Proceedings. 13th International Conference on. 2004.

[6] Carter, C., Hybrid Client-Server, Peer-to-Peer framework for MMOG. in Multimedia and Expo (ICME), 2010 IEEE International Conference on. 2010.

[7] Yan, C., Cooperative peer-to-peer streaming: An evolutionary game-theoretic approach. Circuits and Systems for Video Technology, IEEE Transactions on, 2010. 20(10): p. 1346-1357.

[8] Chu, Y.H., A case for end system multicast. Selected Areas in Communications, IEEE Journal on, 2002. 20(8): p. 1456-1471.

[9] Kuo, W.K. and S.Y. Lien, Dynamic resource allocation for supporting real-time multimedia applications in IEEE 802.15.3 WPANs. Communications, IET, 2009. 3(1): p. 1-9.

[10] Unapproved IEEE Draft Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 17: Resilient Packet Ring (RPR)Access Method and Physical Layer Specifications (Superseded by 802.17-2004). IEEE Std P802.17/D3.3, 2004.

[11] IEEE Draft Amendment to Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications IEEE Draft Amendment to - Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 3: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications - Media Access Control Parameters, Physical Layers and Management Parameters for Subscriber Access Networks (Amendment to 802.3-2002). IEEE Std P802.3ah/D3.3, 2004.

[12] Wang, J., Statistical time division multiplexing based local system architecture for multi-cluster NoC. in Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on. 2011.

[13] Shoaran, M., Compact Low-Power Cortical Recording Architecture for Compressive Multichannel Data Acquisition. Biomedical Circuits and Systems, IEEE Transactions on, 2014. 8(6): p. 857-870.

[14] Quintana-Ramirez, I., Optimization of P2P-TV traffic by means of header compression and multiplexing. in Software, Telecommunications and Computer Networks (SoftCOM), 2013 21st International Conference on. 2013.

[15] Joos, P. and W. Verbiest. A statistical bandwidth allocation and usage monitoring algorithm for ATM networks. in Communications, 1989. ICC '89, BOSTONICC/89. Conference record. 'World Prosperity Through Communications', IEEE International Conference on. 1989.

[16] Kramer, G., B. Mukherjee, and G. Pesavento, IPACT a dynamic protocol for an Ethernet PON (EPON). Communications Magazine, IEEE, 2002. 40(2): p. 74-80.

[17] Janowski, L. and Z. Papir. Analysis of a burstiness curve for FBM traffic and token bucket shaping algorithm. in Telecommunications, 2005. ConTEL 2005. Proceedings of the 8th International Conference on. 2005.

[18] Valenzuela, J.L., A hierarchical token bucket algorithm to enhance QoS in IEEE 802.11: proposal, implementation and evaluation. in Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th. 2004.

[19] Aeron, A. Fine Tuning of Fuzzy Token Bucket Scheme for Congestion Control in High Speed Networks. in Computer Engineering and Applications (ICCEA), 2010 Second International Conference on. 2010.

[20] Chang-Hwan, L. and K. Young-Tak. QoS-aware hierarchical token bucket (QHTB) queuing disciplines for QoS-guaranteed Diffserv provisioning with optimized bandwidth utilization and priority-based preemption. in Information Networking (ICOIN), 2013 International Conference on. 2013.