BANDWIDTH SCHEDULING ALGORITH FOR MULTIPLE DATA TRANSFERS IN HIGH PERFORMANCE NETWORKS (HPNS)

BY

AMAD UR REHMAN (144449)

KASIITA TAWFIK (144450)

MOHAMED UMARU (144447)

SUPERVISOR

PROF.DR.MUHAMMAD MAHBUL ALAM

BADWIDHTH SCHEDULING IN ALGORITHMS FOR MULTIPLE DATA TRANSFER IN HIGH PERFORMANCE NETWROKS (HPNS)

APPROVED BY:

____

PROF.DR MUHAMMAD MAHBUB ALAM

SUPERVISOR AND PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, ISLAMIC UNIVERSITY OF TECHNOLOGY

DATE ________________

Table of Contents

Acknowledgment

A report with the objectives requires a lot of effort to be prepared. But all these efforts can be proven fruitless without proper guidance and adequate source of information. Since the project report involves implementing textual concepts into real life problems. It require the late two criteria the most.

To guide our report any closer to the objectives, we would at first like to thank our supervisor Mr. Md Mahbub Alam Professor and Head of the CSE Department of Islamic University of Technology (IUT). He has on several appointments with us provided us the strategic focuses to bisect the existing problem of interest and from there to reach to a solution for that,

Secondly, we would like to thank Assistant Professor Ashraf Alam khan Department, Islamic University of Technology (IUT) for his kind cooperation in providing us different sorts of information as well as guidance.

Lastly, we would like to thank the staff of CSE Department. They were very cordial with us throughout the year and provided all the lab equipment timely.

ABSTRACT

The significance of high-performance dedicated networks has well recognized due to the rapidly increasing number of large-scale applications that require high-speed data transfer. Efficient algorithms are needed for path computation and bandwidth scheduling in dedicated networks to improve the utilization of network resources and meet diverse user request. We consider three periodic bandwidth scheduling problems: multiple data transfer allocation (MDTA), multiple data transfer allocation with shortest job first (MDTA/SJF) and multiple fixed-slot bandwidth reservation (MFBR), all of which schedule a number a number of user requests accumulated in a certain period. MDTA is to assign multiple data transfer requests on several pre-specified network paths to minimize the total data transfer end time, while MDTA/SJF is to minimize the total transfer end time by sorting the user requests in increasing order, while MFBR is to satisfy multiple bandwidth reservation requests, each of which specifies a bandwidth and a time slot. For MDTA, and MDTA/SJF we design an optimal algorithm and provide its corresponding proof. For MFBR we prove an algorithm and purpose of the algorithm, Minimal Bandwidth and Distance Product Algorithm (MBDPA). Extensively simulation results illustrate the performance of superiority of the proposed MDBPA algorithm over a greedy approach and provide valuable insight into the advantage of periodic bandwidth scheduling over instant bandwidth scheduling.

Chapter 01: Introduction

1.1 Introduction

Many large-scale applications in science and business domains require the transfer of big data over high-performance networks for remote operations.

Such big data transfer is increasingly supported by bandwidth reservation services that discover feasible and efficient routing options in dynamic network environments with time-varying resources.

Applications in various domains such as e-science, e-business, and social media are now producing large amounts of data on a daily basis, which must be transferred over wide geographical areas for various remote operations.

Typical examples include next generation computational sciences where large simulation datasets produced on supercomputers are shared by a distributed team of scientists for collaborative visualization and analysis.

Fast and reliable data transfer has become a critical task to ensure the success of these applications.

Unfortunately, the sheer volume of data generated in such applications has gone far beyond the capability of traditional shared IP networks.

In recent years, high-performance networks (HPNs) that provision dedicated channels through bandwidth reservation have emerged as a promising solution and their significance has been well recognized in broad science and network research communities for big data transfer.

As the central function unit of a generalized control plane for provisioning dedicated channels in HPNs, the bandwidth scheduler computes appropriate network paths and allocates link bandwidths to meet specific user requests based on network topology and bandwidth availability.

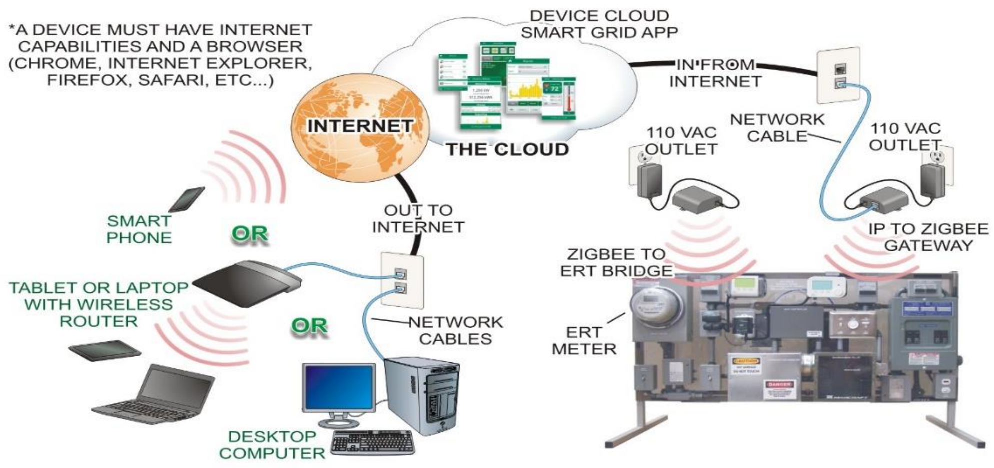
The bandwidth scheduler in HPNs computes appropriate network paths and allocates link bandwidths to meet specific user requests based on network topology and bandwidth availability.

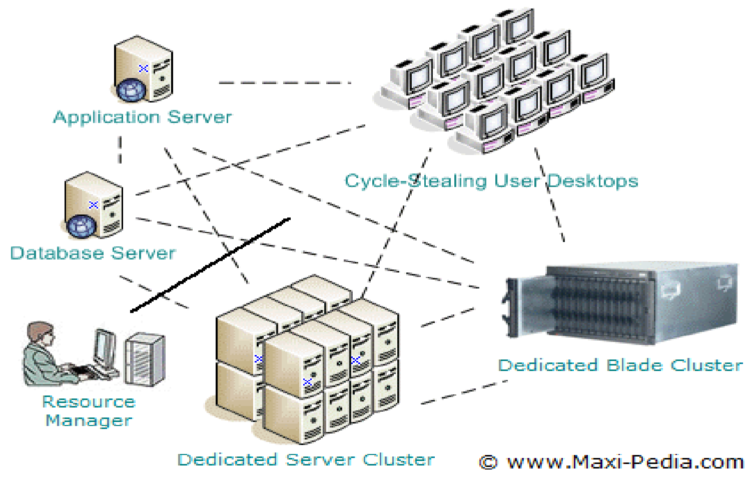
To meet the unprecedented requirement of big data movement, it is a natural extension from single-path to multi-path transfer, which is generally more effective in terms of throughput, robustness, load balance, and congestion reduction.

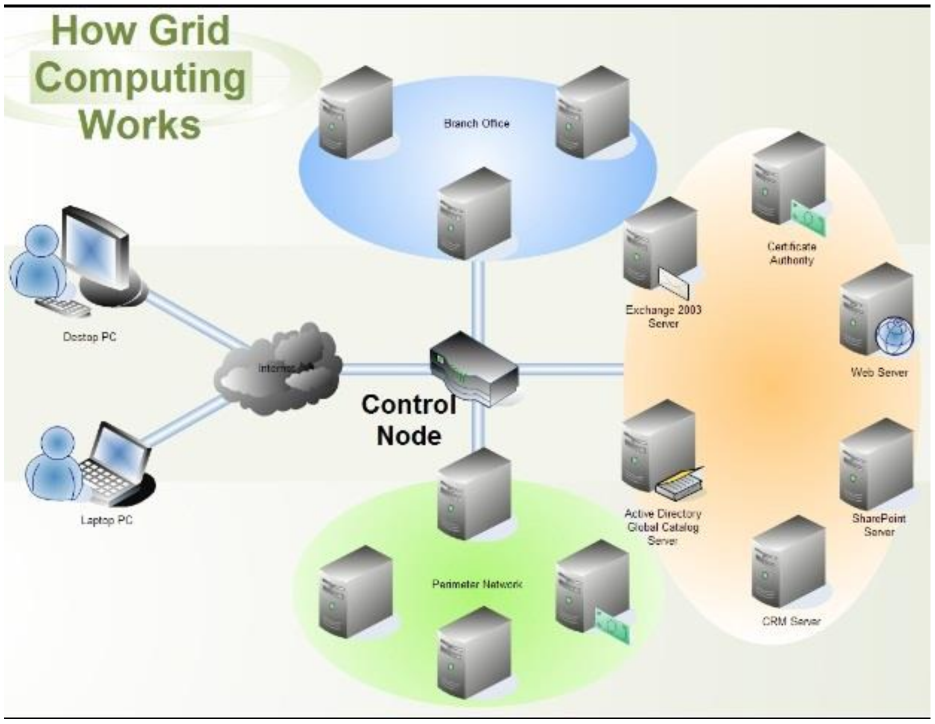
Therefore it is a natural extension from single-path to multi-path transfer, which could be either link-disjoint or node-disjoint.

Particularly, node-disjoint paths can establish multiple independent data channels between source and destination, and hence can effectively increase transmission bandwidth and reliability.

However, multi-path routing also introduces extra overhead to both the control plane and the data plane of a network


*A DEVICE MUST HAVE INTERNET CAPABILITIES AND A BROWSER (CHROME, INTERNET EXPLORER, FIREFOX, SAFARI, ETC...)
DEVICE CLOUD SMART GRID APP
INTERNET
THE CLOUD
IN FROM INTERNET
NETWORK CABLE
110 VAC OUTLET
110 VAC OUTLET
SMART PHONE
OR
OUT TO INTERNET
IP TO ZIGBEE GATEWAY
ZIGBEE TO ERT BRIDGE
TABLET OR LAPTOP WITH WIRELESS ROUTER
OR
NETWORK CABLES
ERT METER
DESKTOP COMPUTER


Application Server
Cycle-Stealing User Desktops
Database Server
Dedicated Blade Cluster
Resource Manager
Dedicated Server Cluster
© www.Maxi-Pedia.com

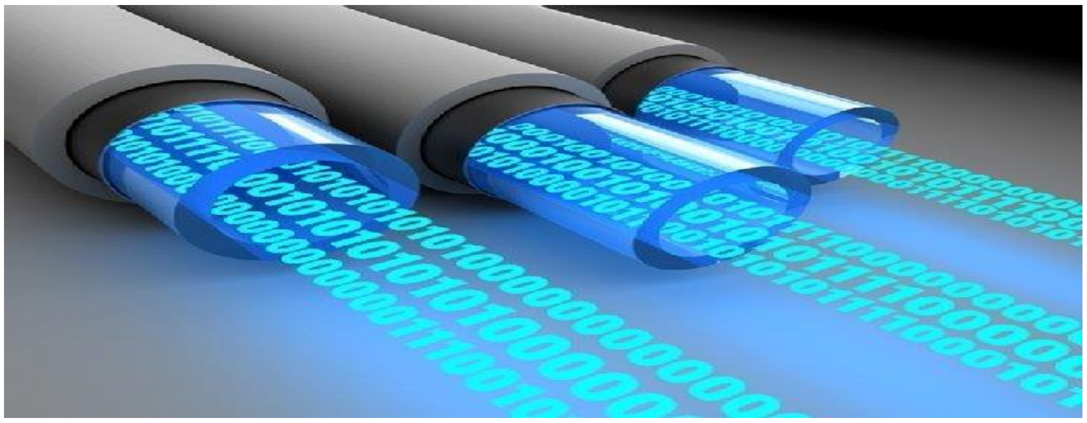

How Grid Computing Works
Branch Office
Certificate Authority
Exchange 2003 Server
Desktop PC
Internet
Web Server
Control Node
Active Directory Global Catalog Server
SharePoint Server
Laptop PC
Perimeter Network
CRM Server

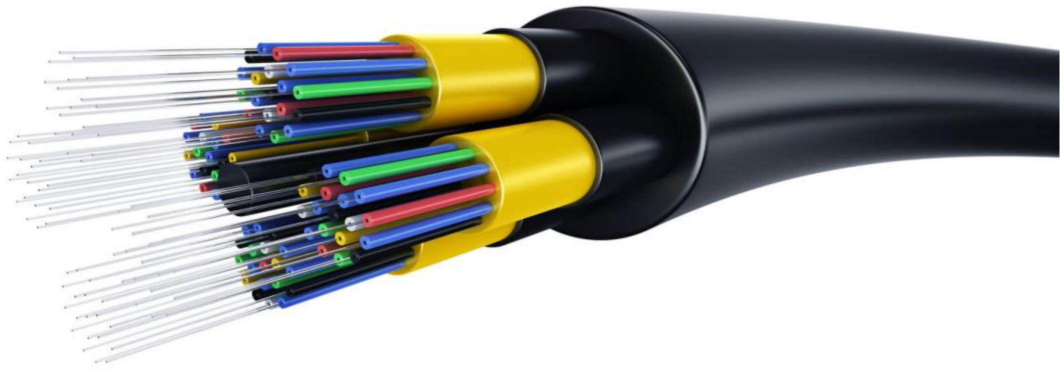
1.2 What is Bandwidth?

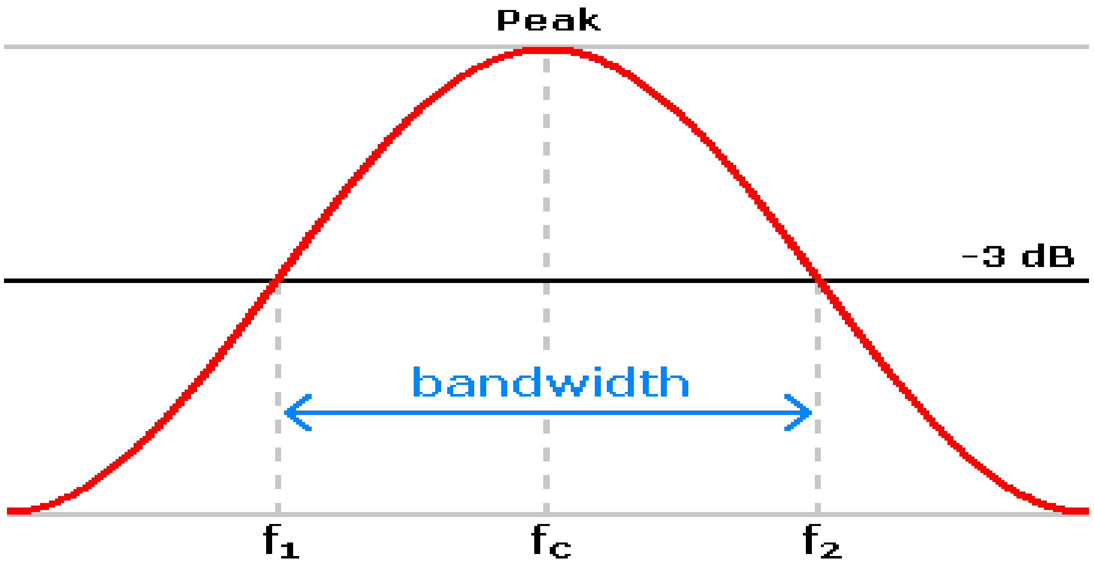
There are many definitions of bandwidth that exist. In computing, bandwidth is the maximum rate of data transfer across a given path.

Bandwidth may be characterized as network bandwidth, data bandwidth, or digital bandwidth.

Bandwidth is defined as a range within a band of frequencies or wavelengths. Bandwidth is also the amount of data that can be transmitted in a fixed amount of time.






Peak
-3 dB
bandwidth
f_1
f_c
f_2

1.3 What is Scheduling?

In computing, scheduling is the method by which work specified by some means is assigned to resources that complete the work.

The work may be virtual computation elements such as threads, processes or data flows, which are in turn scheduled onto hardware resources such as processors, network links or expansion cards.

A scheduler is what carries out the scheduling activity. Schedulers are often implemented so they keep all computer resources busy (as in load balancing), allow multiple users to share system resources effectively, or to achieve a target quality of service

Scheduling is fundamental to computation itself, and an intrinsic part of the execution model of a computer system; the concept of scheduling makes it possible to have computer multitasking with a single central processing unit (CPU).

A scheduler may aim at one or more of many goals, for example: maximizing throughput (the total amount of work completed per time unit); minimizing wait time (time from work becoming enabled until the first point it begins execution on resources); minimizing latency or response time (time from work becoming enabled until it is finished in case of batch activity, or until the system responds and hands the first output to the user in case of interactive activity); or maximizing fairness (equal CPU time to each process, or more generally appropriate times according to the priority and workload of each process). In practice, these goals often conflict (e.g. throughput versus latency), thus a scheduler will implement a suitable compromise. Preference is measured by any one of the concerns mentioned above, depending upon the user's needs and objectives.

In real-time environments, such embedded systems for automatic control in industry (for example robotics), the scheduler also must ensure that processes can meet deadlines; this is crucial for keeping the system stable. Scheduled tasks can also be distributed to remote devices across a network and managed through an administrative back end.

1.4 High Performance Networks

Healthcare networks and member expectations are changing. Increasingly, employees are concerned about more than the size of their network. Instead, they want a healthcare network that offers it all: high-quality physicians, reasonable costs, and a streamlined claims process. For many employers, the best option might be a high-performance network like Canopy Health.

1.5 What is a high-performance healthcare network?

High-performance networks focus on quality rather than quantity. They are a form of narrow network (one that offers fewer physician options, typically less than 25% of a community's providers). In exchange for a smaller network, members pay lower premiums. A high-performance network carefully selects its physicians based on the quality of their care and other metrics.

1.6 Why are high-performance networks increasing in popularity?

In 2017, about 15% of employers offered health plans with a high-performance network. Another 9% offered a plan with a narrow network. However, narrow networks are more popular on healthcare exchanges. In 2016, more than 80% of healthcare plans on the California marketplace offered narrow networks. A high-performance or narrow network can cost up to 35% less than a traditional network.

Chapter # 03: Our Work

3.1 INTRODUCTION:

A number of large-scale application in various science engineering and business domain are generating colossal amounts of data, on the order of terabyte currently and petabytes in the near future, which must be transferred over wide geographical areas for remote operations. Typical examples include next generation computational science applications where large simulation data sets produced on super computers are shared by a distributed team of collaborative scientists [1, 2, 3]. Since the data providers and consumers in these distributed applications are generally located at different sites across the nation or around the globe, high-speed dedicated connections are needed to support a variety of remote tasks including data mining, consolidation, alignment, storage, visualization and analysis [13]. Providing the capability of large data transfer over dedicated channels is critical to ensuring the success of these applications.

The significance of high-performance networks has been well recognized in the broad science and network research communities, and several projects are currently underway to develop such network capabilities, including User Controlled Light Paths(UCLP)[4], UltraScience Net(USN)[18], Circuit-Switched High-Speed End-to-End Transport Architecture(CHEETAH)[16], Enlightened [5], Dynamic Resource Allocation via GMPLS Optical Networks(DRAGON)[6]m Japanese Gigabit Network[7], Bandwidth on Demand (BoD) on Geant2 Network[8], On-demand Secure of Circuits and Advance Reservation System(OSCARS) [9] of ESnet, Hybrid Optical and Packet Infrastructure(HOPI)[10], Bandwidth Brokers [38], and other networks. Such dedicated channels are a part of the capabilities envisioned for Global Environment of Network Innovations (GENI) project [11]. Furthermore, such deployments are expected to increase significantly and proliferate into both public and dedicated network infrastructures across the globe in the coming years. An evidence of this trend in production networks is reflected by internet offering on-demand circuits and Multiple Protocol Label Switching (MPLS) tunnels, and ESnet offering dedicated Virtual Local Area Networks (VLAN) using OSCARS.

The hardware infrastructure including edge devices, core switches, and backbone routers in these high-performance networks are generally coordinate by a management framework, namely control plane, which is responsibility for allocating link bandwidth to users, setting up end-to-end transport paths upon request, and releasing resources when task are completed. As the central function unit of a generalized control plane for provisioning dedicated channels, the bandwidth scheduler computes appropriate network paths and allocated links bandwidths to meet specific user requests based on the networks topology and bandwidth availability. Hence, the performance of the bandwidth scheduler has a significant impact of the utilization of network resources and the productivity of end users.

To achieve high throughput over dedicated channels, a number of transport methods have been developed based on either TCP enhancement or UDP with non AIMD source rate control. These transport provide a variety of transport capabilities ranging from maximizing link utilization, stabilizing throughput at a fixed target rate, to aggregating multiple data streams along different transport paths.

In order to improve the utilization of network resources and meet diverse user request, we consider two periodic bandwidth scheduling problems: multiple data transfer allocation (MDTA) and multiple data transfer allocation with shortest job first (MDTA/SJF). We would like to point out that different from instant scheduling algorithm executed in a certain interval to schedule a number of user requests accumulated during that period. Specifically, MDTA is to assign multiple data transfer requests on several pre-specified network paths to minimize the total data transfer end time. Note that the transfer end time consists of both transfer time and waiting time. A real life network example of using MDTA is to schedule the transfer of a large number of data sets between two remote sites where core switches are deployed and connected with multiple parallel dedicated link. A practical application scenario using MTDAB/SJF is to establish several control channels between collaborative sites for computational monitoring and steering operations that typically require smooth and stable data flows with constant bandwidth during certain time slots.

In dedicated networks that support in advance bandwidth provisioning, the existing bandwidth allocations on a link in future time slots are typically specified as segmented constant functions. The residual bandwidth on certain links are to be allocated to establish new dedicated several links and matching their bandwidths in corresponding time slots. We will focus on bandwidth scheduling and path computation algorithms that can be applied across connection oriented networks.

3.1.1 BACKGROUND

With the rapid development of dedicated networks, various algorithms for in advance bandwidth scheduling have been proposed in the literature. We provide below a survey of background literature related to our work on the design of bandwidth scheduling algorithms.

Instant scheduling problems have been extensively studied in various network context and many scheduling techniques have been proposed. In [13], the author describe four scheduling, including specified bandwidth in a specified time slots, and all available time slots with specified bandwidth and duration, highest available bandwidth in a specified time slots, and all available time slots with a specified bandwidth and duration. The first three problems are straightforward extensions of the classical Dijkstra's algorithm, while the last algorithm is based on an extension of the Bell-Ford algorithm. Similar problems are also discussed in [14] with a detailed description on the solution to each of the problems. These basic scheduling problems with several extensions are investigated in [17] with a focus on increasing flexibility of services. The scheduling algorithm proposed by Cohen et al in [17] considers the flexibility of transfer start time and capability of path switching between different paths during a connection to improve the network utilization. In [16], Grimmell et al. formulate a dynamic quickest path problem, which deal with the transmission of a message from a source to a destination with the minimum end-to-end delay over a network with the propagation delays and dynamic bandwidth constraints on the links.

Although network researchers are increasingly realizing the importance of scheduling multiple bandwidth requests to improve utilization of expensive network resources, periodic scheduling problems in dedicated networks have not received as much attention as instant scheduling problems. However, there is a great deal of similar work in other networking subjects including optical burst switching and traffic engineering. In optical burst switching, the header of a burst is sent in advance of the data burst to reserve a wavelength channel at each optical switching node along the path. The scheduling problem addressed in our work differ from the work done. In MDTA problem, each file cannot be split among paths during transfer and must be strategically allocated in it's entirely to one multiple predefined paths to minimize the total waiting time. The MDTA/SJF problem is an extension of the fixed slot bandwidth reservation problem in [14], which schedule multiple fixed slot bandwidth reservations to maximize the number of successful bandwidth reservations.

3.2 CONTROL PLANE FRAMEWORK

Now we consider a generalized control plane to support in advance reservation of dedicated channels over high-speed networks. The control plane framework shown in fig.1 consists of the following components:

a. **Client interface**.
b. **Server front-end.**
c. **User management.**
d. **Token management.**
e. **Database management.**
f. **Bandwidth scheduler.**
g. **Signaling daemon.**

The interaction between these components take place either over the data plane or control plane to accomplish the tasks of user specified bandwidth reservation, path computation and networking signaling.

Depending on the system configuration, the control plane operation can be coordinated by a central management node or by a set of node distributed over a network. A browser or a web client, for example SOAP (Simulation Object Access Protocol)-based XML message exchange. Accordingly, the service that accept bandwidth reservation requests from user with valid credentials. The user

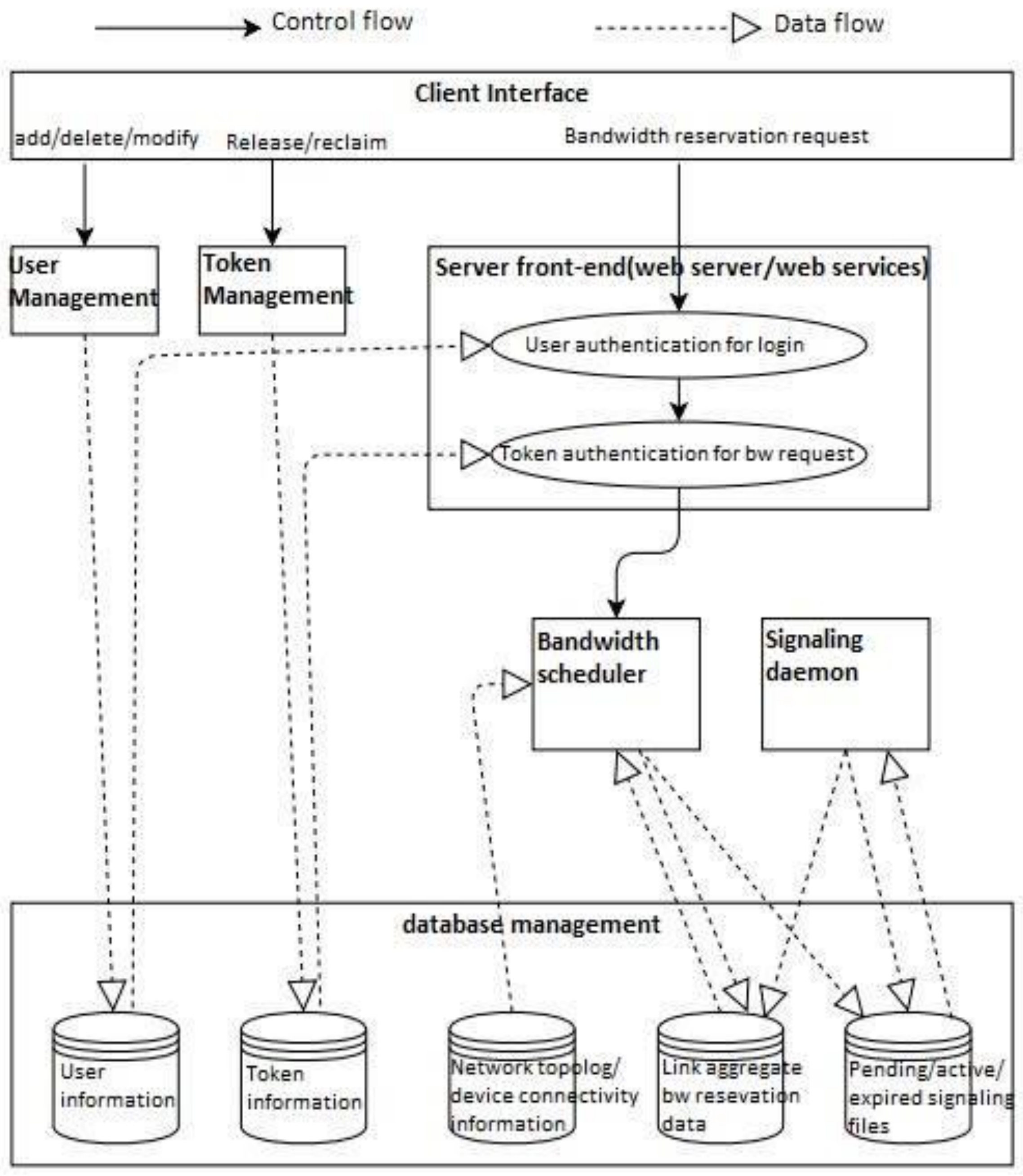

Control flow
Data flow
Client Interface
add/delete/modify
Release/reclaim
Bandwidth reservation request
User Management
Token Management
Server front-end(web server/web services)
User authentication for login
Token authentication for bw request
Bandwidth scheduler
Signaling daemon
database management
User information
Token information
Network topolog/ device connectivity information
Link aggregate bw resevation data
Pending/active/ expired signaling files

Figure: Framework of Control Plane: Function components, control plane, Control flow, and Data flow.

management module supports a special group of users with administrative privileges to add, delete or modify user account information.

User sites are connected through their assigned ports on the edge switches of the network infrastructure. A token based scheme is used for authorization and coordination of channel of setup. Multiple tokens are provided to users for their assigned ports, which they can release to other users and reclaim them as needed. A channel reservation request is honored only if the tokens at both ends are either owned by or released to the user making the request. It is implicitly assumed that users at both end will work out their connectivity mechanisms and policies before the tokens are released.

As the central components of a control plane, the bandwidth scheduler computes one path or set of paths and allocates appropriates links bandwidth to satisfy user data transfer requests. Upon the completion of path computation, a signaling record is generated and bandwidth allocation of each link along that paths is updated accordingly. The signaling daemon periodically examines active or expiring signaling records. For each active or expiring signaling record, the daemon invokes appropriate signaling scripts to set up or tear down the connections along the computed or establish path, respectively. The aggregate bandwidth reservation data for each component link is updated if the signaling actions are successful. Note that the time interval at which the signaling daemon is periodically activated must be chosen to be compatible with the finest resolution of the bandwidth reservation time.

3.3 PERIODIC SCHEDULING PROBLEM FORMULTON

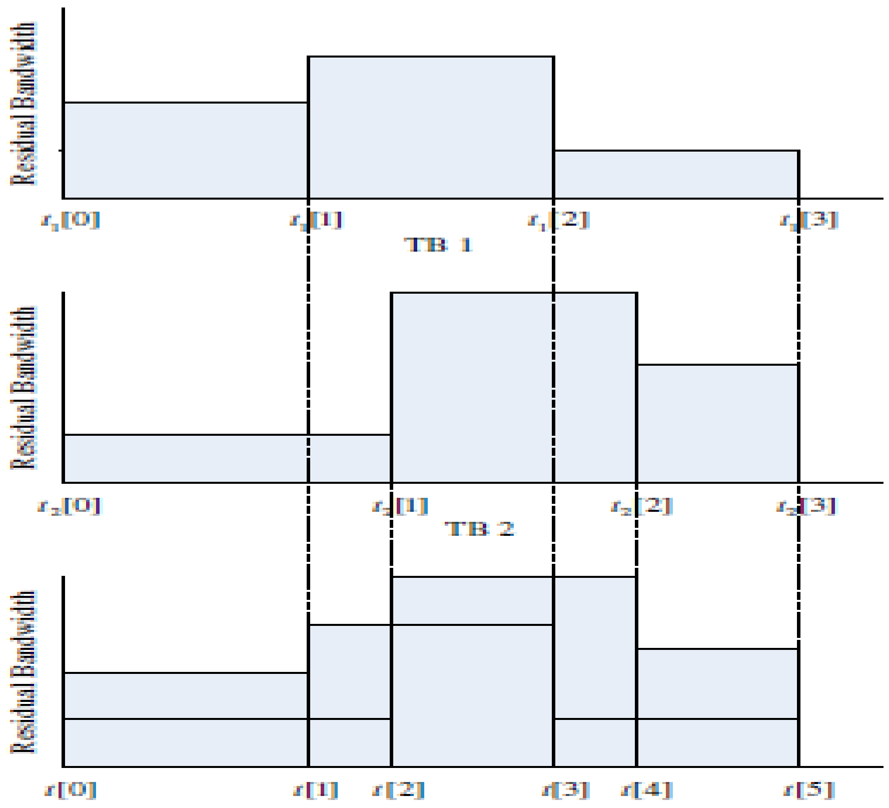
In our periodic scheduling problem formulation we use a graph G= (V,E) to represent the topology of a dedicated network where each link l ϵ E maintains a list of residual bandwidths specified as segmented constant function of time. A pair of time bandwidth (TB) (t$t_{[i]}, b_{[bi]}$) denotes the residual bandwidth of link l at time slot for the that $(t_{[i],}\ t_{[i+1]})$, where i = 1, 2 ,……., T_l where T_l is the total number of time slots on link l.

A path is defined as an ordered set of vertices from the source to destination over one or more links. Before performing the path computation, we combine the time-bandwidth lists of all links to construct an aggregated time-bandwidth list as shown in fig 2. To obtain an aggregated time-bandwidth list, we first create a new set of time slots by combining the time slots of all links$l \epsilon E$, and then map the residual bandwidth of each link to aggregated TB list is denoted as $t[1], b_1[1], b_2[1], \dots \dots . b_m[1], \dots \dots \dots (t[T], b_1[T], b_2[T], \dots \dots b_m[T])$, as now where T is the total number of new time slots and m is the number of links in network G.

We consider two periodic multiple data transfers scheduling problems here in this section.

1. Multiple data transfer allocation bandwidth (MDTA) to assign multiple data transfer request on several specified network paths for the minimum total data transfer end time.
2. Multiple data transfer allocation bandwidth with shortest job first (MDTA/SJF) to where we will schedule such data which is in the increasing form of the data to minimum total data transfer time and also for the purpose of the utilizing the bandwidth in efficient and effective way.

   These problems are formally defined as follows.



Aggregated TB

3.4 MULTIPLE DATA TRANSFER ALLOCATION (MDTA)

In this Algorithm when the user request for the Bandwidth reservation/allocation will first interact with the user request for Bandwidth request. Now for the purpose of the keeping information of user. It have to keep the information of the user in the user management data base. This data base will keep all the information related to the user. The token management will always be used for keeping the token to active user and after finishing the task it will release the token and will regain it when it need it. After getting the user authentication and token management process then it will forward the request to the bandwidth scheduler. The bandwidth scheduler will assign the bandwidth according to the user request and also it will computes the path from the network topology. The network topology will help in keeping the information about the network structure or we can say the topology.

Now after authenticating the user and keeping the information related to the user as well as assigning the bandwidth and path computation. The algorithm will now proceed with the requests it has already and to transfer the multiple data or files from source to destination so that it can minimize the end time delay that is end-to-end time. Now selecting the path and transferring the data on one path or multiple paths it will be based on network structure which is provided by the network topology database system.

To process multiple data from source to destination we have given P pre specified paths from source vertex s to a destination vertex d with an residual bandwidth b and k files of size $\delta_1, \delta_2, \dots \dots ., \delta_k,$ find a task assignment scheme that allocates k files on p paths to minimize the total file transfer end time. In MDTA problem, the user whose primarily interest is in completing the file transfer task as soon as possible, does not need to specify the bandwidth and time slot for each task.

3.5 MULTIPLE DATA TRANSFER ALLOCATION BANDWIDTH WITH SHORTEST JOB FIRST (MDTA/SJF)

We design an optimal algorithm based on an extension of the Shortest Job First algorithm for MDTA, which takes as input p disjoint paths from source to destination d with identical bandwidth b and k files of sizes $\delta_1, \delta_2, \dots \dots., \delta_k,$ and targets minimizing the total transfer end time.

The MDTA/SJF algorithm start with the sorting the files in the increasing order by their sizes, and then take turns to assign each file to one of the p paths. This assignment evenly distributed smaller files onto multiple paths which are transferred before large files. Apparently, transferring a smaller file before a larger one reduces total waiting time, resulting in a shortest total transfer end time. Once the assignment is completed, the files on each path are transferred using the SJF algorithm. The MTDAB/SJF produces the minimal total transfer end time for a given set of files.

To simplify the proof we show a special case where $p = 2$, which can be extended to a general case where $p = 2$. We first consider the files $k = 2n$ of files sorted by their file sizes. The transfer for each file is $t_i = \delta_i / b$, where $t_1 \leq t_2 \dots .. \leq t_{2n}$. Note that the transfer end time includes both waiting time and transfer time. After assigning the $2n$ files to two paths using the MDTA\SJF algorithm, the total transfer end time will be minimize and the bandwidth will use more efficiently and in effective way.

3.5.1 Advantage of MDTA/SJF:

1. The advantage of the MDTA/SJF is that it will help in sorting in increasing order which will help in utilizing the bandwidth in effective way.

2. Once the files is order in increasing order it will minimize the end time delay from source to destination.

.

3.6 Discussion

In this section we will discuss about how to provide the Quality of Service to the user request which will also deal with the minimizing the end time delay. How it will overcome the problems which happens in the previous algorithm. We can provide Quality of Service to the user request by mean of giving it the priority over the other requests so that it can transfer the data over the network more efficiently. We design the algorithm for that which will work based on the previous algorithm which we discuss in the last sections. But here we give the priority to the user request for transferring the data. We will call this algorithm as the multiple user request with Quality of Service (Qos) that is MFBR.

3.6.1 MULTIPLE FIXED-SLOT BANDWIDTH RESERVATION (MFBR).

The objective of the MFBR is to maximize the number of the satisfied bandwidth reservation, and hence scheduling the reservation that use less network resources first will return a better result. Since all reservation have same start and end time, the residual bandwidth of link l is the minimal residual bandwidth among all the time slots between the start and end time based onTB_l, $l \in E$ which is a bottleneck. The greedy algorithm takes as input a graph $G = (V, E)$ with bottle neck bandwidth in a given time slot on each link$l$, $l \in E$ and k fixed-slot bandwidth reservations R. We refer to this algorithm as Greedy(G, R). The user will request for a specific bandwidth and since each link has a specified bandwidth. And upon that request he will be given a priority to transfer the data from source to destination. We will first define several notations and operation to facilitate our explanation on the algorithm.

Following are the notations and operations for the algorithms which we discuss below.

DEqueue (R): dequeue the first element in R.

r: a fixed slot bandwidth reservation.

β^r, v_s^r, v_d^r: the specified bandwidth, source vertex, destination vertex of r, respectively.

$E(\beta^r)$: a subset of E, consisting of links whose residual bandwidth less than β^r.

$G - E(\beta^r)$: the operation of removing the links in $E(\beta^r)$ from G.

$b[v]$: the maximum bandwidth of the component links on the path from v_s to v.

S: a set of vertices whose final narrowest paths from the source have already been determined.

Q: a min-priority queue of vertices, keyed by their bandwidth values.

$Extract\ Min(Q)$: the operation of extracting vertex with the minimal bandwidth in Q.

These are the notations and operation in the algorithm which we discussed above and by following these notations and principals we will apply the algorithm based on the data to be transferred .

3.6.2 Algorithm Design:

The pseudo-code for the algorithm is given in the fig. Now how the algorithm works lest discuss that. So here as we mention earlier the algorithm will transfer the data by giving it the priority to it which means Quality of Service. Here when the user request for the data and it needs the priority over the other. So suppose there are many other files and it need to be transfer in priority way. The algorithm will sort these files in increasing order after one another by their requested bandwidths and schedule them in same order. For each task with a specified bandwidth $\beta,$ the algorithm first removes the links whose residual bandwidth are less than β since these links do not contribute to the current task and subsequent tasks. It then computes the narrowest path in the residual graph by the Dijkstra's algorithm. Here, the narrowest path is defined as the path whose maximum residual bandwidth of all component links on the path from the source vertex to the destination vertex is minimized. Since it will help in minimizing the end time minimization and also utilize the bandwidth in an effective way.

3.6.3 Advantage of the MFBR:

1. It provide Quality of Service to the data.
2. Minimize the end time delay.
3. Utilize the bandwidth in an effective way.

We discuss different algorithms in our work. Each algorithm has advantages and importance. We can use any of the algorithm in order to transfer the data to minimize the end time delay and also utilizing the bandwidth we have in the network in more efficient way. Depending on the user choice and data we can choose the algorithm for transferring the data from the source vertex to the destination vertex. However the benefit of all the algorithms we discussed has the one major advantage that the user can transfer multiple data on the multiple path which will reduce the network burden and traffic contingency.

3.7 Minimal Bandwidth and Distance Product Algorithm (MBDPA):

The scheduling of the tasks largely determines the efficiency of a scheduling algorithm. MBDPA considers the products of bandwidth and distance from source to destination as the amount of network resources needed for each task. Here the distance is counted as the number of hops. The input of MBDPA is the same as the greedy algorithm, and we refer to this algorithm as MBDPA(G, R).

We will define several notations and operations to facilitate our idea.

$d_{(v_s^r, v_d^r)}$: the number of hops from v_s^r to v_d^r.

α^r: the product of the specified bandwidth β^r and $d_{(v_s^r, v_d^r)}$ from a request r.

$ExtractMin(R)$: the operation of extracting the fixed-slot reservation with minimal product α in R.

3.7.1 Algorithm Design:

The pseudo-code for MBDPA(G, R) is shown in algorithm 4. For each fixed-slot reservation r, the algorithm first runs breadth-first search to determine the distance $d_{(v_s^r, v_d^r)}$ from v_s^r to v_d^r in G', which is computed by removing the links with residual bandwidths less than β^r. Here, $d_{(v_s^r, v_d^r)}$ is computed as the number of hops along the found path, and the bandwidth of the path is guaranteed to be at least β^r. Each of the reservation is then keyed by the product of its specified bandwidth and computed distance. The reservation with minimal product is extracted for bandwidth scheduling. The narrowest path with bandwidth β^r is computed by Bellman-Ford algorithm, which iterates for $d_{(v_s^r, v_d^r)}$. The product of each of the remaining tasks is recomputed in each outmost while loop because the network G is updated as bandwidth reservation is recorded at the end of the each loop.

Pseudo-code for Algorithm 1 for MDTA Pseudo-Code:

```
Files of size k
$i = 1; j = 1;$
While($i \leq k$) do
Assign the $i - th$ file in {δ} to join the $j - th$ in {$P$};
$i = 1 + 1;$
$j = 1 + 1;$
If $j > 1;$
end if
end while
```

Pseudo-code for Algorithm 2 for MDTA/SJF:

```
Sort the files in {δ} in an increasing order by sizes;
$i = 1; j = 1;$
While($i \leq k$) do
Assign the $i - th$ file in {δ} to join the $j - th$ in $\{P\}$;
$i = 1 + 1;$
$j = 1 + 1;$
If $j > 1;$
end if
end while
```

Pseudo-code for Algorithm 3 for MFBR:

```
Sort the elements in R by their reserved bandwidth in increasing order;
While R$\neq$ $\emptyset$ do
    $r = Dequeue(R)$;
    $G = G - E(\beta^r)$;
 for all $v = V$ do
      $b[v] = \infty$;
       $prev(v) = Null$;
 end for
 $b[v_s^r] = 0$;
 $S = \emptyset$;
 $Q = V[G]$;
 while $Q \neq \emptyset$ do
        $u = ExtractMin(Q)$;
         $S = S \cup \{u\}$;
         for all $v \in Q, (u, v) \in E$  do
             if $b[v] > \max(b[u], b_{u,v})$ then
                 $b[v] = \max(b[u], b_{u,v})$;
                 $prev(v) = u$;
             end if
         end for
   end while
if $b[v_d^r] = \infty$ then no path can be found to satisfy $r$;
    else
      construct the path from $v_s^r$ to $v_d^r$ and reserve the bandwidth $\beta^r$ on links along the path for r;
    end if
end while
```

Pseudo-code for Algorithm 4 for MBDPA:

```
While R$\neq \emptyset$ do
  for all $r \in R$ do
      $G' = G - E(\beta^r)$;
      compute $d_{(v_s^r, v_d^r)}$ in $G'$ by breadth-first search, $d_{(v_s^r, v_d^r)}$= 0 if no path is found;
  end for
 $r = ExtractMin(R)$;
 for all $v \in V$ do
      $b[v] = \infty$;
      $prev(v) = Null$;
end for
 $b[v_s^r] = \beta^r$;
 for $(i = 1; i \leq d_{(v_s^r, v_d^r)}; i++)$ do
      for all$(u, v) \in E$ do
          if $b[v] > \max(b[u], b_{u,v})$ and $b_{u,v} \geq \beta^r$ then
              $b[v] > \max(b[u], b_{u,v})$;
              $prev(v) = u$;
          end if
      end for
  end for
 if $b[v_d^r] = \infty$ then
      no path can be found to satisfy r;
 else
      Construct the path form $v_s^r$ to $v_d^r$, and reserve the bandwidth $\beta^r$ on links along the path for $r$;
 end if
end while
```

Chapter # 04: Results and Evaluation

4.1 Comparison of Greedy and MBDPA

We conduct performance evaluation of Greedy and MBDPA algorithms for problem solving using simulating networks and bandwidth reservation requests. Each simulated network has an arbitrary network topology with a given number of nodes and links. The residual bandwidths of the links are randomly selected within certain range. We generate a number of fixed-slot reservations whose source and destination vertices are randomly selected and bandwidth are randomly specified within a certain range.

Given the same set of reservations and network configuration, we compare the number of satisfied reservations that are achieved by Greedy and MBPDA under various network topologies and different numbers of bandwidth reservations. The simulation results show that MBDPA consistently outperforms Greedy in all the cases we studied. Since the larger networks sizes are expected to have more satisfied reservations, which is confirmed by the observation in Table 1. For visual comparison purpose, we plot in Fig the performance of the two algorithms under a small network of 10 nodes and 40 links, where only a small portion of 400 reservation are satisfied. We also plot in Fig their performance under a large network of 500 node and 2000 links, where MBDPA algorithm satisfied the majority of the reservation while Greedy algorithm satisfied less than half of the reservation.

Case #	# of nodes (n), # of links (m)	10 tasks		20 tasks		50 tasks		100 tasks		200 tasks		300 tasks		400 tasks	
		G	M	G	M	G	M	G	M	G	M	G	M	G	M
1	n=10, m=40	10	10	19	20	35	46	44	58	57	7777	61	84	64	89
2	n=20, m=80	10	10	20	20	42	45	52	66	64	88	69	105	76	119
3	n=50, m=200	10	10	20	20	45	50	53	78	76	110	90	134	95	150
4	n=100, m=400	10	10	20	20	47	50	83	93	97	135	106	177	120	186
5	n=200, m=800	10	10	20	20	50	50	94	97	122	185	142	227	158	255
6	n=500, m=2000	10	10	20	20	50	50	99	100	169	199	194	285	197	349

Table 1

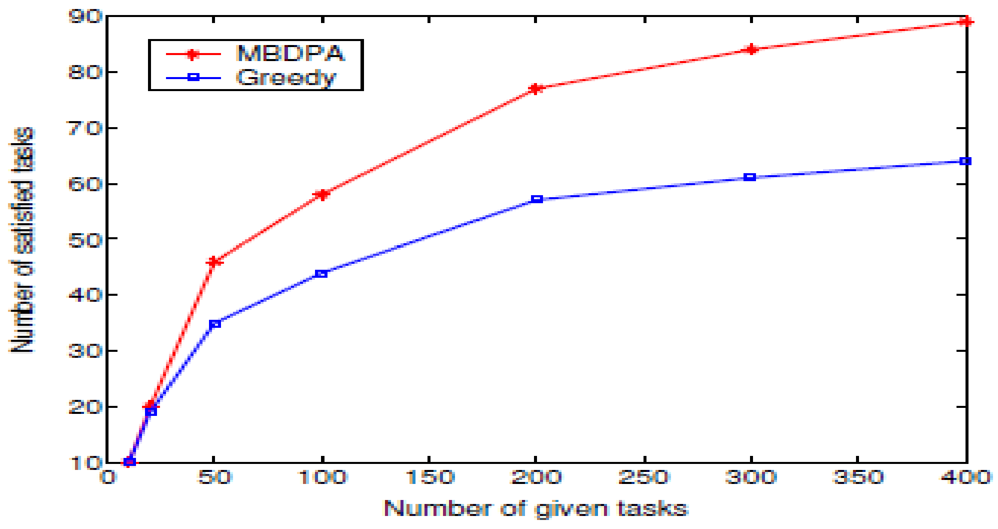


Figure: Comparison of Greedy and MBDPA under the network with 10 nodes and 40 links

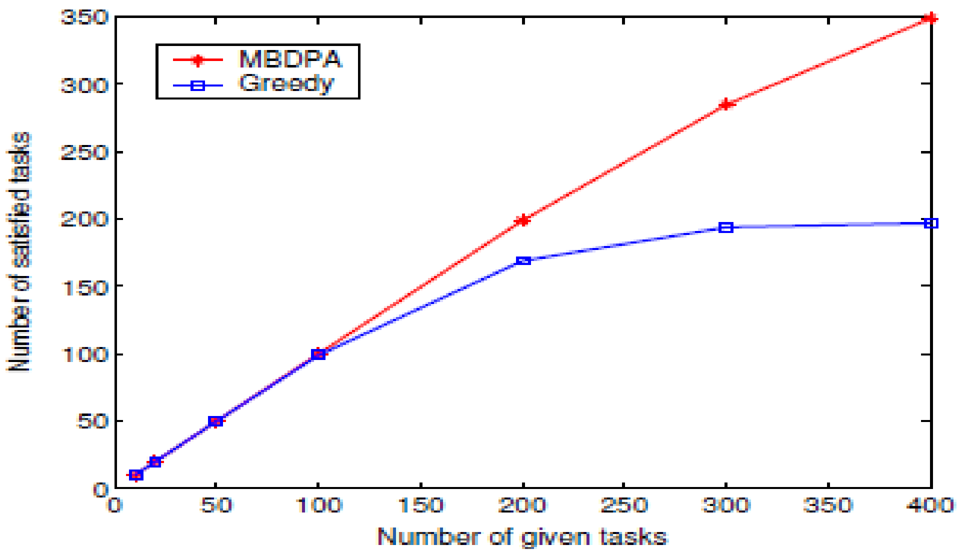


Figure: Comparison of Greedy and MBDPA under the network with 500 nodes and 2000 links

4.2 Periodic Scheduling vs. Instant Scheduling

We compare the performances of periodic scheduling and instant scheduling in both MDTA and MFBR problems to justify the motivation for developing periodic scheduling algorithms. In instant scheduling, tasks are schedule at their arrival times in the arriving order; while in periodic scheduling, scheduling algorithms are launched periodically at a certain time interval on a number of tasks accumulated in one period. In fact, instant scheduling is a specials case of periodic scheduling when the time intervals is set to one time unit. Therefore the performance of periodic scheduling is at least good as instant scheduling if an appropriate scheduling interval is applied. Note that to large degree, the performance of periodic scheduling is determined by scheduling interval, which is confirmed by our simulations described below.

For MDTA problem, based on a number of randomly generated tasks with sizes and arrival times evenly distributed at a given range, we run instant and periodic scheduling algorithms with different values of scheduling interval and plot their corresponding performance curves in Fig. We observed that the periodic scheduling performance curve exhibits an obvious pattern. Therefore the best value for scheduling interval, can be empirically determined as the valley point to minimize the total transfer end time. Periodic scheduling reduces to instant scheduling when scheduling interval is set to the smallest time unit.

We conduct simulations to investigate how different properties of the given tasks effect periodic scheduling performance. We consider two most important task parameters: the number of tasks and variance of data sizes. In the simulations, we vary the number of tasks and variance of data sizes while fixing the other parameters such as the number of paths and their bandwidths, and compute the performance improvement of periodic scheduling using the optimal scheduling interval over instant scheduling in terms of total transfer end times.

The simulation results are shown in Fig, which illustrates that the total transfer end time of instant scheduling is always greater of equal to that of periodic scheduling. We observed that the performance superiority of periodic scheduling becomes more obvious when the values of these two parameter increases, which is due to the fact that at each time interval a larger number of files with the smallest sizes are scheduled first, hence reducing the waiting time.

For MFBR problem, we investigate how the length of the scheduling time interval affects the performance of periodic scheduling. The simulation results produced by MBPDA algorithm on both periodic and instant scheduling are shown in Fig, under the network of size 100 nodes and 400 links, with 300 randomly generated data transfer tasks. The arrival times of these task are distributed at the time range of [0,100] in Fig illustrated that periodic scheduling always achieves a larger number of satisfied tasks compared to instant scheduling, which is not affected by the length of the scheduling them interval. Furthermore, we observed that the performance of periodic scheduling improves when the scheduling time interval increases, which is due to the fact that tasks requiring less network resources will be always scheduling first.

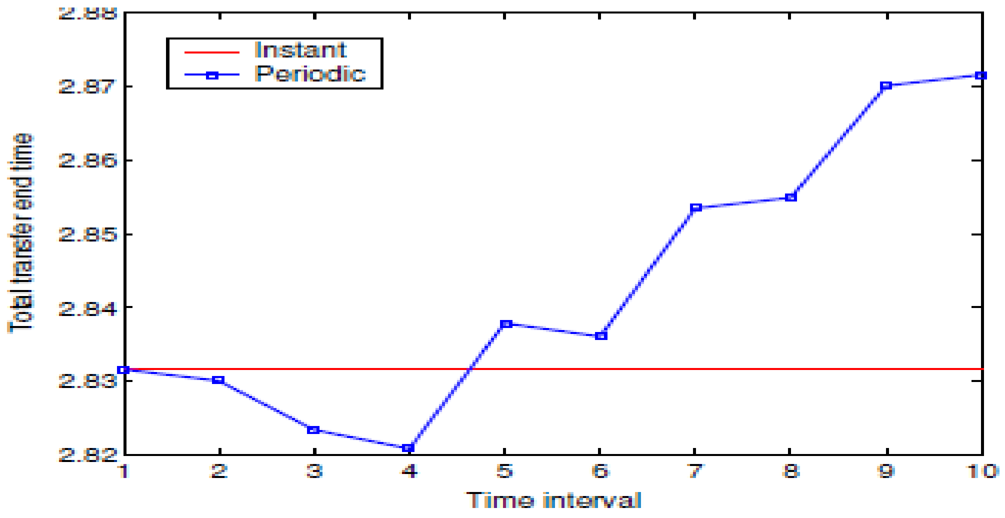


Figure: Comparison of periodic scheduling and instant scheduling in MDTA problem.

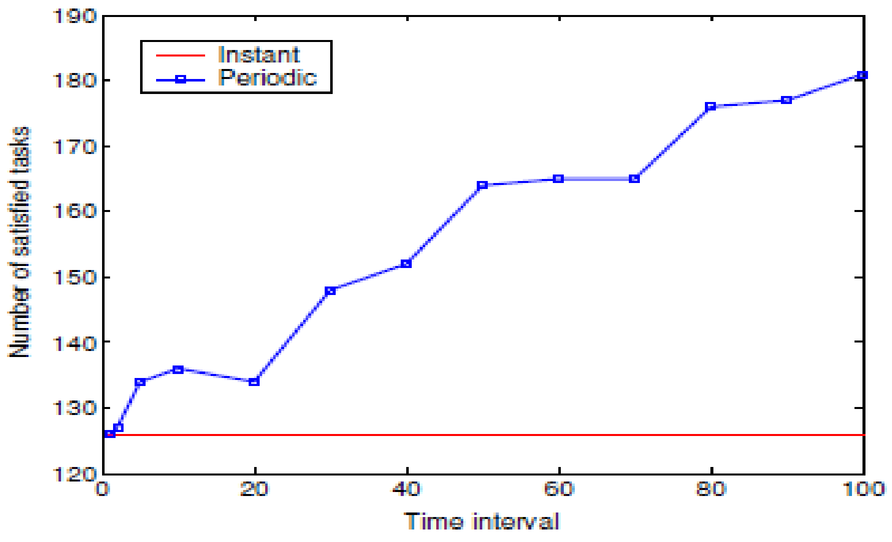


Figure: Comparison of periodic scheduling and instant scheduling in MFBR problem.

Chapter# 05 Conclusion

In this thesis, we considered three periodic scheduling problems, MDTA, MDTA/SJF and MFBR for multiple data transfers in high performance networks. MDTA and MDTA/SJF was solved by an optimal scheduler algorithm. We proved MFBR and purposed of a algorithm, MBDPA, which outperformance the greedy algorithm based on extensively simulation results. The performance superiority of periodic scheduling over instant scheduling justifies our motivation for developing periodic scheduling algorithms to support multiple data transfer requests in High Performance Networks (HPNS).

References

1. Concurrent bandwidth scheduling for big data transfer over a dedicated channel [Liudong Zuo and Michelle M. Zhu*, Chase Q. Wu] Int. J. Communication Networks and Distributed Systems, Vol. 15, Nos. 2/3, 2015.

2. On Design of Scheduling Algorithms for Advance Bandwidth Reservation in Dedicated Networks[Yunyue Lin, Qishi Wu, Nageswara S.V. Rao, Mengxia Zhu] Source: IEEE Xplore May 2008

3. Advance Bandwidth Reservation for Energy Efficiency in High-performance Networks[Tong Shu, Chase Qishi Wu, and Daqing Yun] IEEE 2013.

4. Bandwidth Scheduling and Path Computation Algorithms for Connection-Oriented Networks [Sartaj Sahni Nageshwara Rao Sanjay Ranka Yan Li Eun-Sung Jung Nara Kamath] Sixth International Conference on Networking (ICN'07) 2007.

5. Tera scale High-Fidelity Simulations of Turbulent Combustion with Detailed Chemistry. http://scidac.psc.edu.

6. TerascaleSupernovaInitiative(TSI).http://www.phy.ornl.gov/ti.

7. UCLP:UserControlledLightPathProvisioning.http://phi.badlab.crc.ca/uclp.

8. EnlightenedComputing.http://www.enlightenedcomputing.org.

9. Geant2. http://www.geant2.net.

10. HOPI:HybridOpticalandPacketInfrastructure.http://networks.internet2.edu/hopi.

11. GENI: Global Environment for Network Innovations. http://www.geni.net.

12. Z. Zhang, Z. Duan, and Y. Hou. Decoupling qos control from core routers: A novel bandwidth broker architecture for scalable support of guaranteed services. In Proc. of ACM SIGCOMM, 2000.

13. N. Rao, Q. Wu, S. Carter, W. Wing, D. G.A. Banerjee, and B. Mukherjee. Control plane for advance bandwidth scheduling in ultrahigh-speed networks. In INFOCOM 2006 Workshop on Terabits Networks, 2006.

14. S. Sahni, N. Rao, S. Ranka, Y. Li, E. Jung, and N. Kamath. Bandwidth scheduling and path computation algorithms for connection-oriented networks. In Proc. of Int. Conf. on Networking, 2007.

15. A Scalable and High-Performance Scheduling Algorithm for Multiple Memory Controllers.[Yoongu Kim Dongsu Han Onur Mutlu Mor Harchol-Balter] 2009 IEEE

16. S. Gorinsky and N. Rao. Dedicated channels as an optimal network support for effective transfer of massive data. In INFOCOM 2006Workshop on High-Speed Networks, 2006.

17. R. Guerin and A. Orda. Networks with advance reservations: the routing perspective. In Proc. of the19th IEEE INFOCOM, 2000.

18. N. Rao, W. Wing, S. Carter, and Q. Wu.Ultrascience net: Network test bed for large-scale science applications. IEEE Communications Magazine, 43(11):s12–s17, 2005. An expanded version available at www.csm.ornl.gov/ultranet.

19. Bandwidth scheduling for big data transfer using multiple fixed node-disjoint paths Aiqin Hou, Chase Q. Wu, Dingyi Fang, Yongqiang Wang, Meng Wang.

20. On Design of Scheduling Algorithms for Advance Bandwidth Reservation in Dedicated Networks [Yunyue Lin, Qishi Wu, Nageswara S.V. Rao, Mengxia Zhu] IEEE 2008.

21. https://en.wikipedia.org/wiki/Scheduling_ (computing)

22. https://en.wikipedia.org/wiki/Bandwidth_ (computing)

23. https://www.webopedia.com/TERM/B/bandwidth.html

24. http://www.northropgrumman.com/Capabilities/HighPerformanceNetworking/Pages/default.aspx

25. https://www.canopyhealth.com/en/brokers/articles/high-performance-network.html