A New Approach in Advance Network Reservation and Provisioning for High-Performance Scientific Data Transfers *

Mehmet Balman[1], Evangelos Chaniotakis[2], Arie Shoshani[1], Alex Sim[1]

[1] *Computational Research Division, Lawrence Berkeley National Laboratory*

[2] *Energy Sciences Network, Lawrence Berkeley National Laboratory*

January 2010

Abstract

Scientific applications already generate many terabytes and even petabytes of data from supercomputer runs and large-scale experiments. The need for transferring data chunks of ever-increasing sizes through the network shows no sign of abating. Hence, we need high-bandwidth high speed networks such as ESnet (Energy Sciences Network). Network reservation systems, i.e. ESnet's OSCARS (On-demand Secure Circuits and Advance Reservation System) establish guaranteed bandwidth of secure virtual circuits at a certain time, for a certain bandwidth and length of time. OSCARS checks network availability and capacity for the specified period of time, and allocates requested bandwidth for that user if it is available. If the requested

reservation cannot be granted, no further suggestion is returned back to the user. Further, there is no possibility from the users viewpoint to make an optimal choice. We report a new algorithm, where the user specifies the total volume that needs to be transferred, a maximum bandwidth that he/she can use, and a desired time period within which the transfer should be done. The algorithm can find alternate allocation possibilities, including earliest time for completion, or shortest transfer duration - leaving the choice to the user. We present a novel approach for path finding in time-dependent networks, and a new polynomial algorithm to find possible reservation options according to given constraints. We have implemented our algorithm for testing and incorporation into a future version of ESnets OSCARS. Our approach provides a basis for provisioning end-to-end high performance data transfers over storage and network resources.

Keywords: Dynamic Networks, Graph Theory and Algorithms, On-demand Bandwidth Allocation, Scientific Data Management

1 Introduction

We are witnessing a new era that offers opportunities to conduct scientific research with the help of recent advancements in computational and storage technologies. Computational intensive science spans multiple scientific domains, such as particle physics, climate modeling, and bio-informatics simulations. Scientific applications generate many terabytes and even petabytes of data. In addition to extreme storage requirement, these large-scale applications necessitate collaborators to access very large data sets resulting from simulations performed in geographically distributed institutions. Often, scientific experimental facilities generate massive data sets that need to be transferred to validate the simulation data in remote collaborating sites. For example, in high energy physics, Large Hadron Collider (LHC) is expected to generate 100 gigabits per second in the near future. The generated data is propagated to other research sites for further analysis. Similarly, in the Earth System Grid (ESG), 35 terabytes of data is shared by more than 14000 users worldwide; and the next generation climate

data archive is expected to be more than 1 petabyte.

The need for transferring data chunks of ever-increasing sizes through the network shows no sign of abating. A major component needed to support these needs is the communication infrastructure which enables large-scale data replication, high performance remote data analysis and visualization, and also provides access to computational resources. In order to provide high-speed on-demand data access between collaborating institutions, national governments support next generation research networks such as Internet2 and the Energy Sciences Network (ESnet). Delivering network-as-a-service that provides predictable performance, efficient resource utilization and better coordination between compute and storage resources is highly desirable. Research institutions developed dedicated high-bandwidth networks which bring the ability to provision the communication channels when the data, especially large-scale massive data, is ready to be transferred.

We study network provisioning and advanced bandwidth reservation in ESnet for on-demand high performance data transfers. A reservation request from a user includes desired bandwidth allocation between end-points with duration and starting time information. A bandwidth reservation system, called On-demand Secure Circuits and Advance Reservation System (OSCARS), serves as the network provisioning agent on ESnet. OSCARS checks network availability and capacity for the specified duration of time, and allocates it for the user if it is available. Otherwise, it reports to the user that it is unable to provide the required allocation. Accordingly, it falls upon the user to search for a time-frame of a required bandwidth by trial-and-error, not having knowledge of the network's available capacity at a certain instant of time. We plan to improve the current ESnet advance network reservation system, OSCARS, by presenting to the clients possible reservation options and alternatives for earliest completion time and shortest transfer duration.

In this paper, we present a novel approach for path finding in time-dependent transport networks with bandwidth guarantees. We report an algorithm, where the user specifies the total volume that needs to be transferred, a maximum bandwidth that can be used and provisioned in

the client sites, and a desired time window within which the transfer should be done. The proposed algorithm can find alternate allocation possibilities, including earliest time for completion, or shortest transfer duration - leaving the choice to the user. It is quite practical when applied to large networks with hundreds, even thousands of routers and links. We have implemented our algorithm for testing and incorporation into a future version of OSCARS.

The organization of this paper is as follows. In Section 2, we highlight related work in the literature and compare with our new approach. In Section 3, we explain network reservation, define the problem and give details of a new network reservation service. In Section 4, we present challenges in time-dependent transport networks with bandwidth guarantees. In Section 5, we provide details about our methodology and discuss efficiency of the proposed algorithm. Finally, we conclude with a brief discussion on future work.

2 Related Work

Dedicated bandwidth channels are crucial requirements in distributed computing middleware to satisfy large scale data movement (Li et al. 2008; Rao et al. 2005). On-demand bandwidth circuits provide predictable performance and predictable data transfer duration. Advance network reservation helps users and clients tools in the cooperating organizations to prepare for efficient and fast data movement. This also enables well-organized resource utilization in which communicating parties can plan ahead and provision collaborating resources.

There are few studies in advance bandwidth reservation in the literature (Guerin and Orda 2000; Rao et al. 2006; Burchard 2005). The network reservation problem and path computation with guaranteed bandwidth have been categorized into several domains in (Sahni *et al.* 2007; Jung *et al.* 2008). One of those problems is to reserve a fixed slot in which we find a path from source to destination with a specific bandwidth requirement. Some other cases include finding the path with the largest bandwidth in a specific time slot, and finding the first time slot in which there is a

path with the specified bandwidth requirement from source to destination. Furthermore, bandwidth scheduling problems for multiple data transfer requests are introduced in (Lin & Wu 2008; Veeraraghavan *et al.* 2004; Ganguly *et al.* 2008). The main objective in (Lin & Wu 2008) is to assign a network path for each reservation request with fixed bandwidth in predetermined time period. A greedy heuristic is given in which requests consuming less resource are given preference in scheduling. Those given algorithms have high complexity and large space requirements. They do not compute an optimal reservation for a massive data transfer request, and do not suggest any allocation pattern.

In our approach, we discretize the time-dependent dynamic network topology by dividing the search interval into time steps. Each time step represents a stable status of the topology. We provide a methodology to calculate static snapshot graphs in each time steps and and apply max-bandwidth algorithm while traversing over the search interval. We show that the number of subsequent combinations of time steps, the number of time windows, is bounded by the number of reservations in the system. Searching the given time interval is accomplished in polynomial time. Hence, we provide an efficient algorithm to find possible advance network reservation options for the given data transfer requirements.

3 Advance Network Reservation: Background Information

Esnet provides high-bandwidth connections between more than 40 research laboratories and academic institutions for data sharing and video/voice communication. Experimental facilities, supercomputing centers and thousands scientists are connected with ESnet. The ESnet's bandwidth reservation system, OSCARS, establishes guaranteed bandwidth of secure virtual circuits at a certain time, for a certain length of time and bandwidth. Though OSCARS operates within the ESnet, it also supplies end-to-end provisioning between multiple autonomous network domains. OSCARS

gets reservation requests through a standard web service interface, and conducts a Quality-of-service (QoS) path for bandwidth guarantees. Multi-protocol Label Switching (MPLS) and the Resource Reservation Protocol (RSVP) enable to create a virtual circuit using Label Switched Paths (LSP's). It contains three main components: a reservation manager, a bandwidth scheduler, and a path setup subsystem (Guok *et al.* 2006). The bandwidth scheduler needs to have information about the current and future states of the network topology in order to accomplish end-to-end bandwidth guaranteed paths.

The OSCARS bandwidth reservation system keeps track of changes in the network status and maintains a topology graph which can simply be described as follows. Every port in a router has a maximum bandwidth available for reservation, and each network link connecting two ports (providing communication from one router towards another one) has an 'engineering metric' related to the link latency. The engineering metric represents the preferred routing pattern. The web service interface enables users to allocate a fixed amount of bandwidth for a time period between two end-points in the network.

A reservation request R contains source node v^s and destination node v^d, requested bandwidth $\mathtt{M}$, start time t^s and end time t^e: $R = (v^s, v^d, \mathtt{M}, t^s, t^e)$. Since there might be bandwidth guaranteed paths in the system that are already fully or partially committed, the reservation engine needs to ensure availability of the requested bandwidth from source to destination for the requested time interval. In order to eliminate over commitment, committed reservations between start and end times are examined to extract available bandwidth information for each link in the time period. The shortest path is calculated based on the engineering metric on each link, and a bandwidth guaranteed path is set up from source to destination, to commit the reservation request for the given time period.

Problem Definition: Advance network reservation systems like OSCARS enable users to obtain guaranteed requested bandwidth for a certain duration of time. On the other hand, if the requested reservation cannot be granted, no further suggestion is returned back to the user, except a

failure message. In such a situation, users have to go through a trial-and-error sequence, and may need to try several advance reservation requests until they get an available reservation. These try-and-error attempts may also overload the system. Even if a user successfully reserves the network, the choice of requested allocation might not be one of the optimal ones available in the system. Further, there is no possibility from the user's point of view to be aware of the other possibilities that might fit better into his/her requirements. In other words, users cannot make an optimal choice. Moreover, the current method of selecting a path may lead to ineffective use of the overall system such that network resources may not be used as optimally as possible.

Our goal is to enhance the OSCARS reservation system by extending the underlying mechanism to provide a new service in which users submit their constraints and the system suggests possible reservation options satisfying users' requirements.

Network Reservation Engine: We developed a new methodology in which users submit constraints and the system suggests possible reservations options. In this approach, instead of giving all reservation details such as the amount of bandwidth to allocate between start/end times, users provide maximum bandwidth they can use, total size of the data requested to be transferred, the earliest start time, and the latest completion time. Moreover, users can set criteria such that they would like to reserve a path for earliest completion time or reserve a path for shortest transfer duration. Such a request can be represented as: $S = (v^s, v^d, \mathtt{M}^{max}, D, t^E, t^L)$, where D is total size of data to be sent from v^s to v^d, and t^E the earliest start time, t^L is the latest end time. The maximum bandwidth $\mathtt{M}^{max}$ is related to the capability of the client and server hosts between source and destination end-points. Even if the network can provide a higher bandwidth than the maximum requested, the user is not be able to use all the available bandwidth due to limitations and bottlenecks in the client and server sites. The reservation engine finds out a reservation $R = (v^s, v^d, \mathtt{M}, t^s, t^e)$ for the earliest completion or for the shortest duration where $\mathtt{M} \leq \mathtt{M}^{max}$ and $t^E \leq t^s < t^e \leq t^L$.

4 Time-Dependent Transport Networks

In advance network reservation, we first need to ensure the availability of the requested bandwidth before committing a bandwidth allocation request. The foremost question is how to find the maximum bandwidth available for allocation from a source node to a destination node. The max-bandwidth path algorithm (Piotrów 2002) is well known in quality-of-service (QoS) routing problems in which a path is constructed from source to destination whose bandwidth is maximized, given that each link is associated with an available bandwidth value.

The QoS condition is a bottleneck constraint in max-bandwidth path calculation. Alternatively, in shortest path calculation, we find a path whose sum of weights is minimized, and QoS constraint is additive (minimum delay path, or minimum hop count path). The max-bandwidth path algorithm is a slightly modified version of Kruskal and Dijkstra's algorithms with the same asymmetrical time complexity (Piotrów 2002). In the shortest path algorithm, the weight of a path is the sum of values added by each link in the path. On the other hand, the weight of a path in max-bandwidth is the minimum link bandwidth, the bottleneck link over the path. Those algorithms are very fast and efficient, and they have been adapted to deal with many problems in routing and gateway protocols. In a graph with n nodes, there is a total $n!$ paths from source to destination. The main advantage of those types of graph algorithms is that maximum n^2 paths are visited even in worst case.

We deal with a dynamic network such that the bandwidth value for every link is time dependent. While constructing a path and calculating the available bandwidth over a path, we need to consider another variable, time; therefore, the dimension of the problem is extended by adding the time variable such that the state of the topology depends on the time period. Graph algorithms for time-dependent dynamic networks has been studied in the literature especially for max-flow and shortest path algorithms (Orda & Rom 1990; Ding *et al.* 2008; Chabini 1998). The most common approach is the discrete-time algorithms in which the time is modeled as a set of

discrete values and a static graph is constructed for every time interval. As an example, (Cheng *et al.* 2003) uses time-expanded max flow for data transfer scheduling, and (Orda & Rom 1990) presents various shortest path algorithms for dynamic networks with time-dependent edge weights.

Analogous Example: We need different types of algorithms to analyze time-dependent max-bandwidth path calculation. The following is given to clarify the advance bandwidth reservation in dynamic networks. Assume a vehicle wants to travel from city A to city B where there are multiple cities between A and B connected with separate highways. Each highway has a specific speed limit but we need to reduce our speed if there is high traffic load on the road, and we know the load on each highway for every time period. The first question is which path the vehicle should follow in order to reach city B as early as possible. Alternatively, we can delay our journey and start later if the total travel time would be reduced. Thus, the second question is to find the route along with the starting time for shortest travel duration.

Time-dependent graph algorithms mainly focus on those two questions. However, we are dealing with bandwidth reservation where allocation should be set in advance when a request is received. If we apply this condition to the example problem described above, we have to set the speed limit before starting and cannot change that during the journey. Therefore, known algorithms do not fit into our problem domain. This distinguishes our path calculation from other time-dependent graph algorithms in the literature.

5 Methodology and Algorithm

We define the network topology as a time-dependent directed graph $G_T(\mathtt{V}, \mathtt{E}, T)$, with a vertex set $\mathtt{V}$ of n nodes, and an edge set $\mathtt{E} \subseteq \mathtt{V} \times \mathtt{V}$ of m links between nodes. For every edge, $e_k : (v_i, v_j)$, there is a function of available bandwidth $x^{e_k}(t)$ where t is a variable in time domain T. The available bandwidth $x^{e_k}(t)$ in G_T is time-dependent, nonnegative, and bounded by an upper limit u^{e_k}, where u^{e_k} is the maximum bandwidth available for allocation in e_k; such that, $0 \leq x^{e_k}(t) \leq u^{e_k}$ for any instance of

time in T.

When an advance reservation $R_i = (v_i^s, v_i^d, \mathtt{M}_i, t_i^s, t_i^e)$ is confirmed between start time t_i^s and end time t_i^e, we setup a path δ_i from source node v_i^s to destination node v_i^d that can satisfy the allocation of the requested bandwidth $\mathtt{M}_i$. For every edge along the path $\delta_i : (e_{k^i}, e_{k^j}, \dots)$, we allocate $\mathtt{M}_i$ amount of bandwidth for the future use of reservation R_i. The available bandwidth x^{e_k} of each edge in δ_i is updated in the topology graph G_T for the time period of $[t_i^s, t_i^e]$.

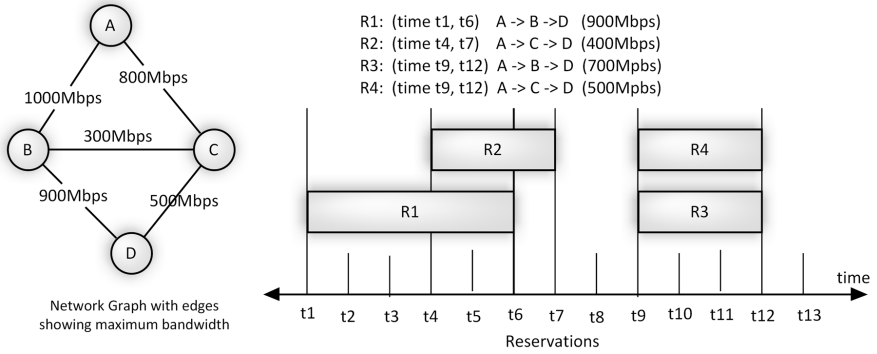

Figure 1: Example for Advance Network Reservation

The example in Figure 1 is given to clarify the underlying mechanism in advance network reservation. At a point of time, assume that there are four reservations confirmed and active in the system; $R_1 = \{A \rightarrow B \rightarrow D, 900Mbps, t_1, t_6\}$, $R_2 = \{A \rightarrow C \rightarrow D, 400Mbps, t_4, t_7\}$, $R_3 = \{A \rightarrow B \rightarrow D, 700Mbps, t_9, t_{12}\}$, $R_4 = \{A \rightarrow C \rightarrow D, 500Mbps, t_9, t_{12}\}$. Thus, the first reservation, R_1, is for 900Mbps between t_1 and t_6 from source A to destination D. The system calculated a path based on engineering metric satisfying requested allocation, and allocated bandwidth over $A \rightarrow B \rightarrow D$. R_2, R_3, and R_4 are interpreted similarly. Figure 2 shows the available bandwidth and allocated bandwidth in link $A \rightarrow B$ over time.

The first graph in Figure 3 represents the status in $[t_1, t_4]$ and the second represents the status in $[t_4, t_6]$.We can confirm a new reservation request from source A to destination D with start time t_1 and end time t_4, with 500Mbps guaranteed bandwidth, and we can allocate path $A \rightarrow C \rightarrow D$ for

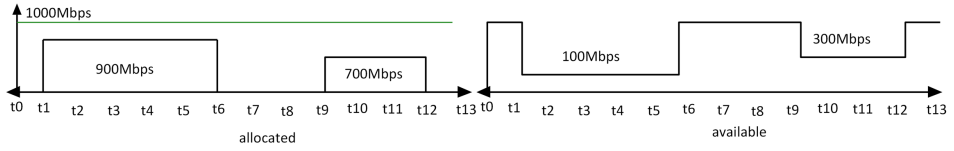

Figure 2: Available bandwidth and allocated bandwidth in link $A \rightarrow B$ over time

the $[t_1, t_4]$ time period. However, we can allocate 100Mbps between t_4 and t_6. Furthermore, we can only allocate 100Mbps between $t1$ and $t6$ because the maximum amount of bandwidth we can get during the entire period of $[t_1, t_6]$ is 100Mbps. Additionally, we cannot split the bandwidth among separate paths. For example, there is an opportunity to send 500Mbps from A to C. The maximum flow from A to C is 500Mbps in $[t_4, t_6]$, 100Mbps over $A \rightarrow B \rightarrow C$ and 400Mbps over $A \rightarrow C$. However, we make a reservation for a specific path. Therefore, the maximum amount of bandwidth we can allocate for a single reservation from A to C is 400Mbps in time period $[t_4, t_6]$.

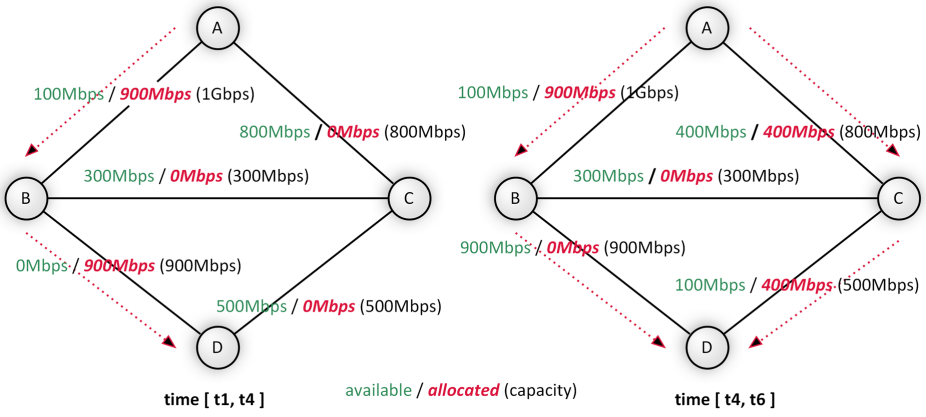

Figure 3: Network Flow in specific time periods($[t_1, t_4]$, $[t_4, t_6]$)

A service request is defined as $S_i = (v_i^s, v_i^d, \mathtt{M}_i^{max}, D_i, t_i^E, t_i^L)$; with total size of data D_i to be sent from v_i^s to v_i^d, and and a period of time between earliest start time t_i^E and latest end time t^L such that, this data

transfer need to be accomplished in this given time interval. If there exists bandwidth between v_i^s and v_i^d within the time constraints in G_T, a new reservation $R_{earliest}$ for earliest completion time or $R_{shortest}$ for shortest transfer duration is generated. Consequently, we create a reservation $R_j = (v_i^s, v_i^d, \mathtt{M}_j, t_j^s, t_j^e)$ where $\mathtt{M}_j \leq \mathtt{M}_i^{max}$ and $t_i^E \leq t_j^s < t_j^e \leq t_i^L$. We also compute a path δ_j satisfying reservation R_j.

In order to satisfy the given criteria, the amount of bandwidth allocation $\mathtt{M}_j$ and the time interval $[t_j^s, t_j^e]$ need to be sufficient to transmit data volume of D_i using the path δ_j allocated for reservation R_j. We can simple say $D_i = \mathtt{M}_j \times d$ where d is the duration between start time t_j^s and end time t_j^e. $R_{shortest}$ has the minimum duration $d = |t^s, t^e|$ among all other possible reservation satisfying S_i. The objective for earliest completion time is to select a reservation R_j satisfying the criteria given in S_i which has the earliest end time t^e. On the other hand, we would favor a reservation with a shorter duration if there are more than one possible reservations completing at the same earliest time. For reservation $R_{earliest}$, $\forall R_j$ satisfying S_i: $t_{earliest}^e \leq t_j^e$, and $\forall R_j$ with $t_j^e = t_{earliest}^e : t_{earliest}^s \geq t_j^s$.

5.1 Search Interval between Earliest-Start and Latest-End times

The outline of our approach is as follows. We divide the given search interval into time steps. The search interval $[t_i^E, t_i^L]$ is the time period between earliest start time t_i^E and latest end time t_i^L in which the data need to be transmitted. A time step represents the longest duration of time in which we have a stable discrete status in terms of available bandwidth over the links. A time period $[t_i, t_j]$ is considered as a time step if $\forall e_k \in G_T : x^{e_k}(t) = c_k$ where $t_i \leq t \leq t_j$, and c_k is a constant. We obtain a static directed graph that keeps information about the available bandwidth status for every link. This information is updated on-the-fly every time a reservation request is committed and stored for further processing during the path calculation phase. A snapshot graph of G_T in time step $ts(t_i, t_j)$ is defined as $G(ts_i)$, with the same vertex set and same edge set. For every edge $e_k : (v_i, v_j)$ in

$ts(t_i, t_j)$, the available bandwidth $x^{e_k} = c_k$ stands for the value of $x^{e_k}(t)$ in G_T between t_i and t_j in time step $ts(t_i, t_j)$. This help us discretized the dynamic graph and apply known graph algorithms efficiently.

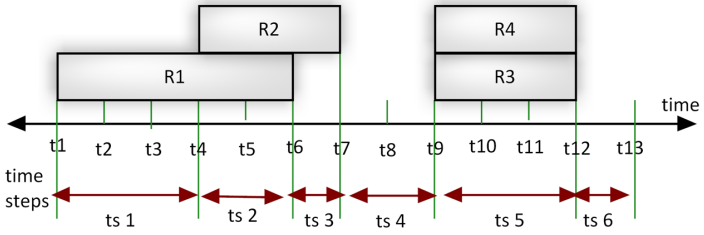

Figure 4: Time steps between t_1 and t_{13}

Figure 4 shows time steps betwen t_1 and t_{13}, for the example given in Figure 1 with four committed reservations. We have six time steps: $ts_1(t_1, t_4)$, $ts_2(t_4, t_6)$, $ts_3(t_6, t_7)$, $ts_4(t_7, t_9)$, $ts_5(t_9, t_{12})$, $ts_6(t_{12}, t_{13})$. Every time step corresponds to a static snapshot of the network topology. Figure 5 shows $G(ts_1)$, $G(ts_2)$, $G(ts_3)$, $G(ts_4)$, $G(ts_5)$, and $G(ts_6)$.

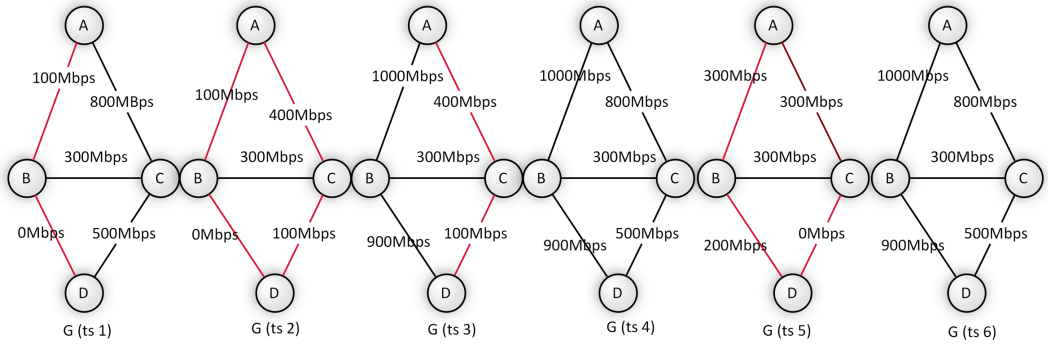

Figure 5: Static Graphs for time steps ts_1, ts_2, ts_3, ts_4, ts_5, ts_6

We analyze the search interval $[t^E, t^L]$ with a set of consecutive time steps covering the entire period. The set of confirmed reservations in the system characterize time steps since they change the available bandwidth values in the network topology. If two reservations partially overlap in terms of time period, they split the total period of time into either two or three time steps. If they do not overlap, they split into three time steps. In other

words, the number of time steps in the search interval is bounded by the number of committed reservations within the given period $[t^E, t^L]$. If there are r committed reservations falling into the period, there can be maximum $2r+1$ different time steps in the worst-case. Figure 4 shows the general idea behind time steps and reservations.

The next step is to traverse these time steps to check whether we can find a reservation satisfying the given criteria. For the example given in Figure 4 and Figure 5, first ts_1, and then ts_2 will be examined; later, if both cannot satisfy the request, time window $tw(t_1, t_6)$, a combination of ts_1 and ts_2, will be examined. A time window consists of subsequent time steps. tw_k is a time window which corresponds to the time period in ts_k. $tw_{k_1-k_2}$ is a time window including all time steps between ts_{k_1} and ts_{k_2}. If there are s time steps in a given search interval, there are $(s \times (s+1))/2$ time windows since time windows are subsequent combinations of time steps.

We search through these time windows in a sequential order to check whether we can satisfy the requested allocation in that time window. For a bandwidth allocation with the shortest duration, we can sort time windows according to their length, and start checking with the smallest one. For a bandwidth allocation with the earliest completion time, we can benefit from a specific search pattern. The search pattern for earliest completion time in the given example will be as follows: tw_1, tw_2, tw_{1-2}, tw_3, tw_{2-3}, tw_{1-3}, tw_4, tw_{3-4}, tw_{2-4}, $tw_{3-4}, \ldots$. The algorithm will stop searching when it finds a time window satisfying the given criteria. In most cases, we do not need to check all possible time windows. In the worst-case, we may require to search all time windows, which makes $(s \times (s+1))/2$ searches, where s is the number of time steps.

5.2 Examining Time Windows to Find Possible Reservations

While checking a time window to verify whether it can satisfy the request, we first look at the total duration of the time window. We know the max bandwidth $\mathtt{M}^{max}$ user can support, and the total size of data D. Therefore, we first determine the duration of a time window and simply ensure whether

this time window is large enough to satisfy the user request. The length of a time window $d = |tw_{k1-k2}|$ should be larger than the minimum amount of time, $D/\mathtt{M}^{max}$, required to transmit data if $\mathtt{M}^{max}$ bandwidth can be allocated.

Then, we calculate the maximum bandwidth available from source v^s to destination v_d in time window tw. We use max-bandwidth path algorithm over static snapshot graph $G(tw)$. $G(tw)$ can easily be computed using snapshots of time steps that form this time window. $G(tw_k) = G(ts_k)$, and $G(tw_{k_1-k_2}) = G(ts_{k_1}) \circ\ G(ts_{k_1+1}) \circ G(ts_{k_1+2}) \cdots \circ G(ts_{k_2})$. We define a new operator, $\circ$, to intersect static snapshot graphs. $G_1 \circ G_2$ forms a new graph with the same vertex and edge set as in G_1 and G_2. For each edge e_k, the available bandwidth is the minimum of x^{e_k} both in G_1 and G_2. Such that, $\forall e_k \in G_1 \circ G_2 : x^{e_k} = min\{x_1^{e_k}, x_2^{e_k}\}$, where $x_1^{e_k}$ is the available bandwidth of e_k in G_1 and $x_2^{e_k}$ is the available bandwidth of e_k in G_2. This property makes the process easy, since we only need to store one graph snapshot for each starting time window; for example, to obtain $G(tw_{1-3})$, we only need $G(tw_{1-2})$ and $G(tw_3)$, $G(tw_{1-3}) = G(tw_{1-2}) \circ G(tw_3)$.

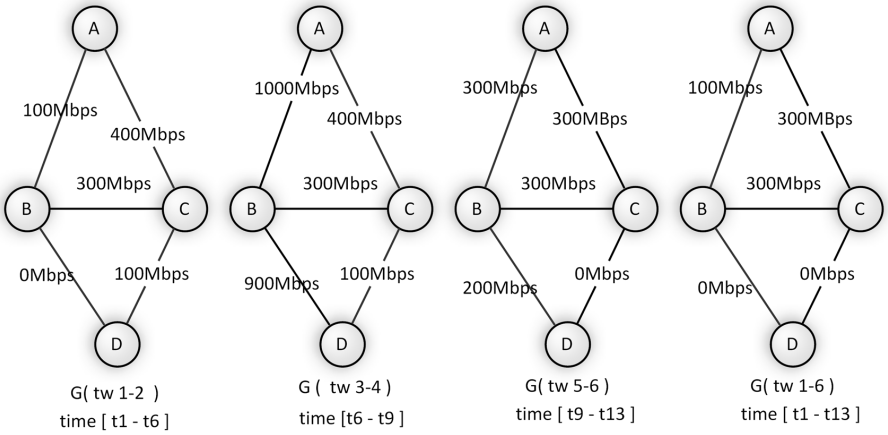

Figure 6: Static Graphs for time windows tw_{1-2}, tw_{3-4}, tw_{5-6}, and tw_{1-6}

Figure 6 shows static snapshot graphs for time windows tw_{1-2}, tw_{3-4}, tw_{5-6}, and tw_{1-6}. $G(tw_{1-2}) = G(ts_1) \circ G(ts_2)$, $G(tw_{3-4}) = G(ts_3) \circ G(ts_4)$, $G(tw_{5-6}) = G(ts_5) \circ G(ts_6)$, and $G(tw_{1-6}) = G(tw_{1-2}) \circ G(tw_{3-4}) \circ G(tw_{5-6})$. R_1 and R_2 are active in time interval $[t_1, t_6]$, so links associated with both

R_1 and R_2 are updated in $G(tw_{1-2})$. Only R_2 is active in time interval $[t_6, t_9]$, so links associated with R_2 are updated in $G(tw_{3-4})$.

While exploring a time window, a max-bandwidth path δ is calculated in $G(tw)$ in which $\mu_{tw}(v^s, v^d)$ is the maximum amount of bandwidth we can allocate in time window tw. $d_{tw} \times \mu_{tw}$ simply gives the amount of data that can be transmitted if a reservation is made in time window tw, where d_{tw} is the length of the time window. A time window $tw(t_i, t_j)$ is selected and marked if it can provide enough resources to satisfy the user criteria. For such a time window, $d_{tw} = |max\{t_i, t^E\}, min\{t_j, t^L\}|$ is the maximum duration we can use to make a reservation, and $\mu_{tw} = \mu_{tw}(v^s, v^d)$ is the maximum amount of bandwidth we can allocate from source to destination. Note that we need to the consider amount of bandwidth we can use which is also limited by the maximum set by the user, $\mu'_{tw} = min\{\mu_{tw}, \mathtt{M}^{max}\}$. Therefore, the product $\mu'_{tw} \times d_{tw}$ should be greater than the requested volume size D.

When a satisfactory window is found, we generate a reservation $R = (v^s, v^d, \mathtt{M}, t^s, t^e)$ and a path from source to destination to be used for this reservation in the network. The start/end times and $\mathtt{M}$ are calculated based on the given user criteria and available resources in the time window. A straightforward strategy to generate a reservation when a time window tw is selected and marked to satisfy the user criteria is as follow: $t^s = max\{t_i, t^E\}$, $\mathtt{M} = min\{\mu_{tw}, \mathtt{M}^{max}\}$, and $t^e = t^s + |D/\mathtt{M}|$.

Input: A set of time steps in the search interval $\{ts_1, ts_2, \ldots, ts_n\}$
Output: A network reservation for earliest completion or shortest duration
for $i = 1$ *to* n **do**
  for $j = i$ *to* 1 **do**
    Get time window $tw = tw_{j-i}$ which contains all time steps between ts_j and ts_i;
    if *the given criteria can fit into the time window* $tw = ts_j \ldots ts_i$ **then**
      Obtain static snapshot graph $G(tw)$ for time window tw;
      Calculate max-bandwidth μ_{tw} from source to destination;
      if *we can satisfy request in time window tw (Examine* μ_{tw}*)* **then**
        select tw ;
    if *goal is to find a reservation with Earliest completion* **then**
      if *there is any selected time window tw* **then**
        Get tw with shortest duration to satisfy the given request;
        Generate a Reservation and a Path, Return for earliest completion;
if *goal is to find a reservation with Shortest duration* **then**
  if *there is any selected time window tw* **then**
    Get tw with shortest duration to satisfy the given request;
    Generate a Reservation and a Path, Return for shortest duration;
Return: No reservation found;
Algorithm: A sample search pattern to find a reservation with earliest completion time or shortest transfer duration

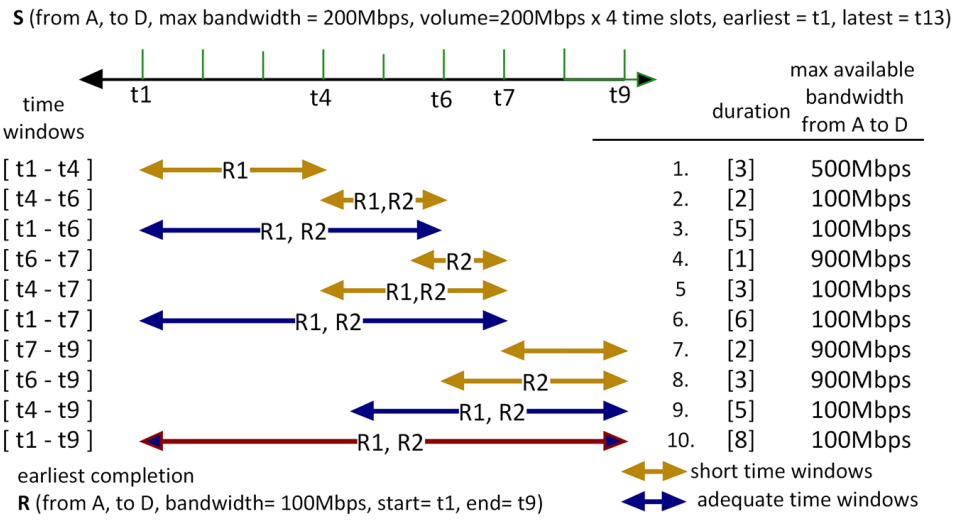

Figure 7: Example for earliest completion

Figure 7 shows the search pattern to find a reservation for the earliest completion time, for the example given in Figure 1. Assume that we have a service request $S = (A, D, 200Mbps, 200 \times 4t, t_1, t_{13})$, and we want to find a reservation satisfying the given criteria. Time window $tw(t_1, t_4)$ with length $3t$, and time window $tw(t_4, t_6)$ with length $2t$, are short in

duration to conform to the requirements of this request. The maximum bandwidth allowed is 200Mbps, so we need at least a time window with length $4t$. $tw(t_1, t_6)$ satisfies the time requirement, so we proceed and calculate the maximum bandwidth available in $G(tw(t_1, t_6))$. The maximum bandwidth we can reserve from A to D between t_1 and t_6 is 100Mbps. Total size of data we can transfer is $100 \times 5t$. Therefore, $tw(t_1, t_6)$ can not satisfy the bandwidth requirement. We keep searching through time windows until we find $tw(t_1, t_9)$ which satisfies both time and bandwidth requirements. Time window $tw(t_1, t_9)$ is selected for the earliest completion time. We generate $R_{earliest} = (A, D, 100Mbps, t_1, t_9)$ with start time t_1 and end time t_9. If we want to find a reservation for the shortest transfer duration, we need to continue searching until we cover the entire interval between t_1 and t_{13}. As shown in Figure 8, $tw(t_9, t_{12})$ and $tw(t_7, t_{12})$, $tw(t_6, t_{12})$, $tw(t_4, t_{12})$, $tw(t_1, t_{12})$, $tw(t_{12}, t_{13})$, $tw(t_9, t_{13})$ ...are searched next. Time window $tw(t_9, t_{13})$ satisfies the given bandwidth and time requirements. All other time windows coming after this in the search pattern, are longer in terms of duration. Therefore, $tw(t_9, t_{13})$ gives the reservation $R_{shortest} = (A, D, 200Mbps, t_9, t_{13})$ with shortest duration. If the total volume of data is $175 \times 4t$, then the search will be same with $R_{shortest} = (A, D, 200Mbps, t_9, t_{12.5})$ and $R_{earliest} = (A, D, 100Mbps, t_1, t_8)$.

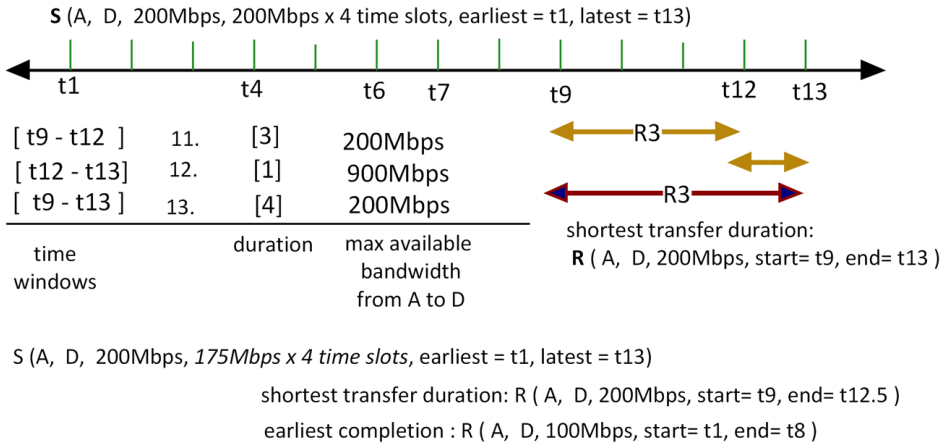

Figure 8: Example for shortest transfer duration

5.3 Evaluation of the Proposed Algorithm

Max bandwidth path algorithm is bounded by $O(n^2)$, where n is the number of nodes in the topology graph. In the worst-case, we may require to search all time windows, $(s \times (s+1))/2$, where s is the number of time steps. If there are r committed reservations in that period, there can be a of maximum $2r+1$ different time windows in the worst-case. Overall, the worst-case complexity is bounded by $O(r^2n^2)$. However, r is relatively very small compared to the number of nodes n, in the topology. Bandwidth reservation is used for large-scale data transfers and it is very unlikely to have thousands of committed reservations in a given time period. Also, the path calculation from two end-points does not span to all nodes in a real network; therefore, we can trim the topology graph and perform calculation on a reduced data set while calculating path from source to destination. Moreover, time windows that are too short in duration to transmit the requested amount of data are eliminated beforehand. Max bandwidth and shortest path algorithms are quite efficient and the search process over time windows is scalable and practical, considering that the number of reservations in practice is limited. Furthermore, there are usually less than a hundred node in a typical network topology like ESnet. We have tested the performance of the algorithm by simulating very large graphs (with 10K nodes) and we have observed that the computation time is in the order of seconds.

6 Conclusion and Future Work

In this study, we presented a new algorithm to find reservation options for earliest completion time and shortest transfer duration. We have also implemented our algorithm and tested with ESnet data. Our goal is to incorporate the algorithm into a future versions of OSCARS. To the best of our knowledge, this algorithm was not previously proposed in the literature.

On the other hand, network provisioning is not sufficient by itself for end-to-end high performance data transfer. In order to take advantage

of the available network bandwidth, client sites should provision other resources such as storage capacity and bandwidth. For this reason, network provisioning services need coordination between storage resource managers, such as SRM (Shoshani *et al.* 2003) that dynamically reserve and manage storage on demand. According to the storage allocation policy and available storage space in client sites, we may need to adjust the network reservation requests. Our future work includes coordination of storage and network resource allocations.

Acknowledgments

This work was funded by the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under contract no DE-AC02-05CH11231.

References

Shoshani, A., Sim, A. & Gu, J. 2003 *Storage Resource Managers: Essential Components for the Grid*. Kluwer Academic Publishers.

Chabini, I. 1998 Discrete dynamic shortest path problems in transportation applications: Complexity and algorithms with optimal run time. *Transportation Research Records*, **1645**, 170–175.

Cheng, W. C., Chou, C., Golubchik, L., Khuller, S. & Wan, Y.-C. J. 2003 Large-scale data collection: a coordinated approach. In *in proceedings of ieee INFOCOM*, pp. 218–228.

Ding, B., Yu, J. X. & Qin, L. 2008 Finding time-dependent shortest paths over large graphs. In *Edbt '08: Proceedings of the 11th international conference on extending database technology*, pp. 205–216. New York, NY, USA: ACM.

Ganguly, S., Sen, A., Xue, G., Hao, B. & Shen, B. 2008 Optimal routing for fast transfer of bulk data files in time-varying networks. IEEE Int. Conf. on Communications.

Guok, C., Robertson, D., Thompson, M., Lee, J., Tierney, B. & Johnston, W. 2006 Intra and interdomain circuit provisioning using the oscars reservation system. In *Broadband communications, networks and systems, 2006. broadnets 2006. 3rd international conference on*, pp. 1–8.

Jung, E.-S., Li, Y., Ranka, S. & Sahni, S. 2008 An evaluation of in-advance bandwidth scheduling algorithms for connection-oriented networks. In *Ispan '08: Proceedings of the the international symposium on parallel architectures, algorithms, and networks*, pp. 133–138.

Lin, Y. & Wu, Q. 2008 On design of bandwidth scheduling algorithms for multiple data transfers in dedicated networks. In *Ancs '08: Proceedings of the 4th acm/ieee symposium on architectures for networking and communications systems*, pp. 151–160.

Orda, A. & Rom, R. 1990 Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. *J. ACM*, **37**(3), 607–625.

Piotrów, M. 2002 A note on constructing binary heaps with periodic networks. *Inf. Process. Lett.*, **83**(3), 129–134.

Sahni, S., Rao, N., Ranka, S., Li, Y., Jung, E.-S. & Kamath, N. 2007 Bandwidth scheduling and path computation algorithms for connection-oriented networks. In *Icn '07: Proceedings of the sixth international conference on networking*, p. 47.

Veeraraghavan, M., Lee, H., Chong, E. & Li, H. 2004 A varying-bandwidth list scheduling heuristic for file transfers. In *Communications, 2004 ieee international conference on*, vol. 2, pp. 1050–1054 Vol.2.

Li, Z., Song, Q., and Habib, I. (2008). Cheetah virtual label switching router for dynamic provisioning in ip optical networks. *Optical Switching and Networking*, 5(2-3):139–149. Advances in IP-Optical Networking for IP Quad-play Traffic and Services.

Guerin, R. and Orda, A. (2000). Networks with advance reservations: the routing perspective. INFOCOMM 2000.

Rao, N., Wu, Q., Ding, S., Carter, S., Wing, W., Banerjee, A., Ghosal, D., and Mukherjee, B. (2006). Control plane for advance bandwidth scheduling in ultra high-speed networks. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–5.

Rao, N. S. V., Wing, W. R., Carter, S. M., and Wu, Q. (2005). Ultrascience net: network testbed for large-scale science applications. *Communications Magazine, IEEE*, 43(11):S12–S17.

Burchard, L.-O. (2005). Networks with advance reservations: Applications, architecture, and performance. *J. Netw. Syst. Manage.*, 13(4):429–449.