Chapter 14
Flexible Bandwidth Scheduling for Streaming Data Movement Over Dedicated Networks

Liudong Zuo, Michelle Zhu, Chase Wu, Nageswara S. V. Rao, Min Han, and Anyi Wang

Abstract A wide range of scientific disciplines are generating large amounts of data at a high speed, which must be transferred to remote sites for real-time processing. Reserving bandwidths over dedicated channels in high-performance networks (HPNs) within a specified time interval has proved to be an effective solution to such high-demanding data transfer. Given a bandwidth reservation request, if the desired bandwidth within the specified time interval cannot be satisfied, most of the existing scheduling algorithms simply reject the request, which would immediately terminate the application. One reasonable approach to mitigate this issue is to provide an alternative bandwidth reservation option to schedule the desired bandwidth within the time interval closest to the specified one. We propose a flexible bandwidth reservation algorithm that considers both the best and alternative bandwidth reservation options for a given request. Extensive

L. Zuo (✉) · A. Wang
Computer Science Department, California State University, Dominguez Hills, Carson, CA, USA
e-mail: lzuo@csudh.edu; awang26@toromail.csudh.edu

M. Zhu
Department of Computer Science, Montclair State University, Montclair, NJ, USA
e-mail: zhumi@montclair.edu

C. Wu
Department of Computer Science, New Jersey Institute of Technology, Newark, NJ, USA
e-mail: chase.wu@njit.edu

N. S. V. Rao
Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN, USA
e-mail: raons@ornl.gov

M. Han
School of Software Engineering, Chengdu University of Information Technology, Chengdu, Sichuan, China
e-mail: hanmin@cuit.edu.cn

© Springer International Publishing AG, part of Springer Nature 2018

N. S. V. Rao et al. (eds.), *Proceedings of International Symposium on Sensor Networks, Systems and Security*, https://doi.org/10.1007/978-3-319-75683-7_14

simulations are conducted to show the superior performance of the proposed scheduling algorithm compared with a heuristic approach adapted from existing scheduling algorithms.

Keywords Bandwidth reservation · Dynamic provisioning · High-performance networks · Quality of service

14.1 Introduction

A wide range of scientific disciplines such as earthquake simulation and high energy physics are generating large amounts of real-time simulation, observational, and experimental data at a high speed [8]. For example, the large-scale data exploration process at the Korea Superconducting Tokamak Advanced Research (KSTAR) generates 3.9 TB of data within only 10 s [3]. Handling such extremely large volumes of data in a timely manner goes far beyond the data processing ability of the current KSTAR. A promising solution for timely data analysis is to quickly move the data to remote collaborating sites from memory to memory, such as Oak Ridge National Laboratory (ORNL) and National Energy Research Scientific Computing Center (NERSC), for near real-time data analysis. How to transfer data at such scales in a fast and reliable way with guaranteed performance is critical for data storage and analysis [9]. Reserving bandwidths over dedicated channels provisioned by high-performance networks (HPNs) within a specified time interval has emerged as a promising solution [1, 4, 6, 12]. For example, the On-Demand Secure Circuits and Advance Reservation System (OSCARS) deployed in ESnet is one of the most extensively used bandwidth reservation services, and ESnet and KSTAR are currently working together to improve the data transfer performance.

In general, a user submits a bandwidth reservation request (BRR) specifying the desired bandwidth to be reserved and the bandwidth reservation start and end time. Upon the receival of a BRR from the user, the bandwidth reservation service provider searches for and allocates corresponding network resources. If there are sufficient available network resources, the desired bandwidth can be successfully scheduled within the specified time interval; otherwise, to the best of our knowledge, most of the existing scheduling algorithms would simply reject the BRR, resulting in an immediate termination of the application. To address this issue, one reasonable approach is to perform flexible scheduling to schedule the desired bandwidth within the time interval closest to the user-specified one and return such option to the user for selection. Accordingly, we propose a bandwidth reservation algorithm, referred to as Flexible Streaming Bandwidth Scheduling (FSBS), which considers both the best and alternative bandwidth reservation options for a given BRR. For performance comparison, we design a heuristic scheduling algorithm adapted from existing scheduling algorithms, referred to as Basic Streaming Bandwidth

Scheduling (BSBS). Extensive simulations are conducted to show the superior performance of FSBS compared with BSBS. To the best of our knowledge, our work is among the first to study bandwidth scheduling with alternative reservation options in HPNs.

The rest of this paper is organized as follows. The related work is described in Sect. 14.2. The mathematical models are presented in Sect. 14.3. The algorithm design and illustration are detailed in Sect. 14.4. The performance evaluations are conducted in Sect. 14.5. We conclude our work in Sect. 14.6.

14.2 Related Work

Big data transfer through bandwidth reservation in HPNs has been widely used in various scientific domains, and many bandwidth reservation problems have been investigated in the past. We conduct a brief survey of related work as follows.

Balman *et al.* proposed one bandwidth reservation algorithm [1] to schedule a user request that specifies the total volume of data to be transferred, a maximum bandwidth that can be used on the client sites, and a desired time interval, within which the transfer must be completed. Upon the receival of such a request, the proposed algorithm finds the bandwidth reservation option with the earliest data transfer completion time or with the shortest data transfer duration. For a similar data transfer request, Lin and Wu considered the following four advance bandwidth scheduling problems: (1) fixed path with fixed bandwidth (FPFB), (2) fixed path with variable bandwidth (FPVB), (3) variable path with fixed bandwidth (VPFB), and (4) variable path with variable bandwidth (VPVB) [6]. The objective is to minimize the data transfer end time for a given transfer request with a pre-specified data size. A detailed problem complexity analysis was conducted, and corresponding algorithms were proposed for each of these problems.

Zuo and Zhu studied the problem of scheduling all BRRs concurrently along different paths in an HPN while achieving their best average transfer performance [14]. These problems were proved to be NP-complete, and heuristic algorithms were proposed. Similar problems on one fixed path were also studied [11, 13]. Zuo *et al.* further studied the scheduling of two generic types of BRRs concerning data transfer reliability: (1) to achieve the highest data transfer reliability under a given data transfer deadline and (2) to achieve the earliest data transfer completion time while satisfying a given data transfer reliability requirement [10, 15]. Two periodic bandwidth reservation algorithms were proposed to optimize the scheduling of individual BRRs within BRR batches.

All of the above work studied the scheduling of bandwidth reservation requests specifying the size of data to be transferred. The problem of bandwidth reservation for streaming data movement still remains open and is the focus of our work, where if the given BRR could not be achieved, we attempt to provide an alternative bandwidth reservation option for the user to choose.

14.3 Mathematical Models

We model an HPN as a graph $G = (V, E)$, where V and E represent the set of nodes and links, respectively [6, 12, 14]. For illustration purposes, we present an example HPN G in Fig. 14.1, where $V = \{v_s, v_0, v_d\}$ and $E = \{v_s - v_d, v_s - v_0, v_0 - v_d\}$. The dynamic bandwidth reservation and release on the links for data movements lead to the variation of G, namely, the available bandwidth of an link $l \in E$ may vary from time to time as shown by the available bandwidth table of links of G on the right side of Fig. 14.1. For convenience, we suppose that link l maintains a list of available bandwidths specified as a segmented constant function of time [6]. We further represent a link's available bandwidth using a 3-tuple of time-bandwidth (TB): $(t_l[i], t_l[i+1], b_l[i])$, which denotes the available bandwidth $b_l[i]$ of link l within time interval $[t_l[i], t_l[i+1]]$, $i = 0, 1, 2, \ldots, T_l - 1$, where T_l is the total number of time slots of link l. We set $t_l[T_l] = +\infty$ as there is no bandwidth reserved on any link of G after time point $t_l[T_l - 1]$. For example, link $v_0 - v_d$ in Fig. 14.1 has three TBs, $(0, 3s, 12Gb/s)$, $(3s, 8s, 10Gb/s)$, and $(8s, +\infty, 12Gb/s)$, while link $v_s - v_d$ only has one TB, $(0, +\infty, 6Gb/s)$.

Bandwidth reservation for data movement is generally made on one network path, which is defined as an ordered set of nodes from the source to the destination by concatenating one or multiple links [6]. Before computing the path for bandwidth reservation, we combine the TB lists of all links together to build an aggregated TB (ATB) list to store the available bandwidths of all links of G in each intersected time slot. The ATB list is denoted as $\{(t[0], t[1], b_0[0], \ldots, b_{|E|-1}[0]), \ldots, (t[T-1], t[T], b_0[T-1], \ldots, b_{|E|-1}[T-1])\}$, where T is the total number of new time slots after the aggregation of the TB lists of all links. For example, after aggregating the TB list of all links of G in Fig. 14.1, we have four time slots: $[0, 3s]$, $[3s, 6s]$, $[6s, 8s]$, and $[8s, +\infty)$, and the ATB list is $\{(0, 3s, 6Gb/s, 9Gb/s, 12Gb/s), (3s, 6s, 6Gb/s, 8Gb/s, 10Gb/s), (6s, 8s, 6Gb/s, 11Gb/s, 10Gb/s)$, and $(8s, +\infty, 6Gb/s, 11Gb/s, 12Gb/s)\}$. For convenience, we put the two endpoints of all time slots in a TreeSet to

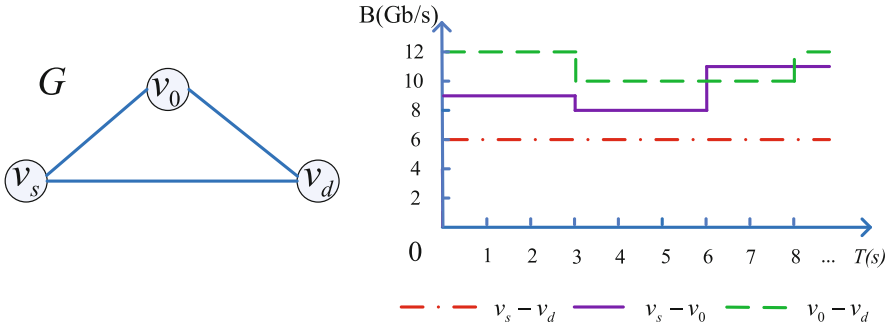

Fig. 14.1 The topology of an example HPN (left) and the available bandwidth table of the links of the example HPN (right)

facilitate the design of scheduling algorithms in Sect. 14.4. For example, the TreeSet after the ATB list aggregation of the links of G in Fig. 14.1 is $\{0, 3s, 6s, 8s, +\infty\}$.

We denote a BRR as a 5-tuple (v_s, v_d, B, t_S, t_E), where the user requests to reserve bandwidth B within time interval $[t_S, t_E]$ from source v_s to destination v_d. For example, suppose that G in Fig. 14.1 receives a BRR at time point 0 requesting to reserve bandwidth of $9Gb/s$ within the range $[4s, 7s]$ from v_s to v_d. The corresponding BRR is denoted as $(v_s, v_d, 9Gb/s, 4s, 7s)$.

We denote a bandwidth reservation option as a 4-tuple (p, B, t_s, t_e), where the HPN schedules bandwidth B on path p within time interval $[t_s, t_e]$. For example, for BRR $(v_s, v_d, 9Gb/s, 4s, 7s)$, we are not able to successfully schedule $9Gb/s$ within $[4s, 7s]$. The bandwidth reservation option that is closest to the required time interval $[4s, 7s]$ is $(v_s - v_0 - v_d, 9Gb/s, 6s, 9s)$, as shown in next section.

14.4 Algorithm Design and Analysis

This section focuses on the algorithm design of FSBS and BSBS. We first present the pseudocode of FSBS and BSBS, followed by the algorithm analysis and a brief illustration using one example BRR.

14.4.1 Algorithm Design and Illustration of FSBS

The pseudocode of FSBS is shown in Algorithms 1–3. We provide a brief illustration of FSBS using the example BRR received by G at time point 0: $(v_s, v_d, 9Gb/s, 4s, 7s)$.

Following Line 1 of Algorithm 1, we build the ATB list and create the TreeSet TS: $\{0, 3s, 6s, 8s, +\infty\}$. After Lines 2–7 of Algorithm 1, the available bandwidths of the links of G are: $b_{v_s-v_d} = 6Gb/s$, $b_{v_s-v_0} = 8Gb/s$, and $b_{v_0-v_d} = 10Gb/s$. The path with the largest available bandwidth returned by the modified Dijkstra's algorithm is $v_s - v_0 - v_d$, and its available bandwidth is $8Gb/s < 9Gb/s$. Hence, the requested bandwidth could not be scheduled within the specified time interval. We create bandwidth reservation list LBR and call Algorithm 2.

Currently, $m' = 1$ and the "while" loop begins. After the iteration, we could not identify any path with available bandwidth of at least $9Gb/s$. The "while" loop continues and $m' = 0$. When $i == 0$, after Lines 4–8 of Algorithm 2, the available bandwidths of the links of G are: $b_{v_s-v_d} = 6Gb/s$, $b_{v_s-v_0} = 9Gb/s$, and $b_{v_0-v_d} = 12Gb/s$. The path with the largest available bandwidth returned by the modified Dijkstra's algorithm is $v_s - v_0 - v_d$, and its available bandwidth is $9Gb/s$. We create bandwidth reservation option $(v_s - v_0 - v_d, 9Gb/s, 0, 3s)$ and add it to LBR. We have $t' = 4s - 3s = 1s$. The "while" loop ends here, and we then call Algorithm 3.

Currently, $n = 3$ and the "while" loop begins. After the iteration, we could not identify any path with available bandwidth of at least $9Gb/s$. The "while" loop

Algorithm 1: FSBS (Flexible Streaming Bandwidth Scheduling)

INPUT: $G = (V, E)$, a BRR (v_s, v_d, B, t_S, t_E).
OUTPUT: One bandwidth reservation option.

1: Combine the TB lists of all links together to build the ATB list. Create a TreeSet TS containing the two endpoints of all time slots. Identify the index of the largest element in TS that is no larger than t_S and the index of the smallest element that is no less than t_E, denoted by m and n, respectively;
2: **for** each $l \in E$ **do**
3: $b_l = +\infty$;
4: **for** $m \leq i \leq n-1$ **do**
5: $b_l = min\,(b_l, b_l[i])$;
6: **end for**
7: **end for**
8: Run modified Dijkstra's algorithm to identify the path with the largest available bandwidth from v_s to v_d within time interval $[TS[m], TS[n]]$. Suppose that the returned path is p and its available bandwidth is b;
9: **if** $b \geq B$ **then**
10: Create bandwidth reservation option (p, B, t_S, t_E) and return it to the user.
11: **else**
12: Create bandwidth reservation option list LBR and call Algorithm 2;
13: Return the bandwidth reservation option in LBR to the user.
14: **end if**

Algorithm 2: Left Closest Bandwidth Reservation Option Computation

INPUT: $G = (V, E)$, TS, BRR (v_s, v_d, B, t_S, t_E), and bandwidth reservation option list LBR.
OUTPUT: NULL.

1: Initialize variables $t' = +\infty$ and $m' = m$;
2: **while** $m' \geq 0$ **do**
3: **for** $n-1 \geq i \geq m'$ **do**
4: **for** each $l \in E$ **do**
5: $b_l = +\infty$;
6: **for** $m' \leq j \leq i$ **do**
7: $b_l = min\,(b_l, b_l[j])$;
8: **end for**
9: **end for**
10: Run modified Dijkstra's algorithm to identify the path with the largest available bandwidth from v_s to v_d within time interval $[TS[m'], TS[i+1]]$. Suppose that the returned path is p and its available bandwidth is b;
11: **if** $(TS[i+1] - TS[m']) \geq (t_E - t_S)$ && $b \geq B$ **then**
12: Add bandwidth reservation option $(p, B, TS[i+1] - (t_E - t_S), TS[i+1])$ to LBR;
13: $t' = t_S - TS[i+1]$ and break the "while" loop;
14: **end if**
15: **end for**
16: $m' - -$;
17: **end while**
18: Call Algorithm 3.

Algorithm 3: Right Closest Bandwidth Reservation Option Computation

INPUT: $G = (V, E)$, TS, the BRR (v_s, v_d, B, t_S, t_E), t', and bandwidth reservation option list LBR.
OUTPUT: NULL.

```
1: while n ≤ |TS| − 1 do
2:   for m ≤ i ≤ n − 1 do
3:     if TS[i] − t_E ≥ t' then
4:       Continue;
5:     end if
6:     for each l ∈ E do
7:       b_l = +∞;
8:       for i ≤ j ≤ n − 1 do
9:         b_l = min(b_l, b_l[j]);
10:      end for
11:    end for
12:    Run modified Dijkstra's algorithm to identify the path with the largest available
       bandwidth from v_s to v_d within time interval [TS[i], TS[n]]. Suppose that the returned
       path is p and its available bandwidth is b;
13:    if (TS[n] − TS[i]) ≥ (t_E − t_S) && b ≥ B then
14:      if t' < +∞ then
15:        Remove the element from LBR;
16:      end if
17:      Add bandwidth reservation option (p, B, TS[i], TS[i] + (t_E − t_S)) to LBR;
18:      Break the "while" loop.
19:    end if
20:  end for
21:  n++;
22: end while
```

continues and $n = 4$. When $i == 2$, after computation, the path with the largest available bandwidth returned by the modified Dijkstra's algorithm is $v_s - v_0 - v_d$, and its available bandwidth is $10Gb/s > 9Gb/s$. Currently, $t' = 1s < +\infty$, we remove the current bandwidth reservation option $(v_s - v_0 - v_d, 9Gb/s, 0, 3s)$ in LBR from LBR and add the newly created bandwidth reservation option $(v_s - v_0 - v_d, 9Gb/s, 6s, 9s)$ to it. The "while" loop ends here.

At this point, LBR contains the following bandwidth reservation option: $(v_s - v_0 - v_d, 9Gb/s, 6s, 9s)$, which is returned to the user. The user makes a decision based on the current situation to either choose or reject the returned bandwidth reservation option. Figure 14.2 shows the topology of G if the user chooses the returned bandwidth reservation option; otherwise, G keeps the same topology as shown in Fig. 14.1.

In the worst case, the time complexity of FSBS is $O(T^3 \cdot |E| + T^2 \cdot (|E| + |V|log|V|))$.

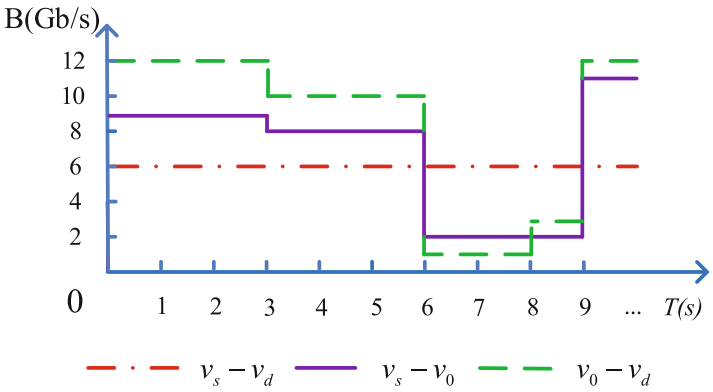

Fig. 14.2 The topology of the example HPN after the user accepts the returned bandwidth reservation option $(v_s - v_0 - v_d, 9Gb/s, 6s, 9s)$

14.4.2 Algorithm Design and Illustration of BSBS

As mentioned in Sect. 14.1, if there is no sufficient resource to satisfy the required bandwidth within the specified time interval, most of the existing bandwidth scheduling algorithms would simply reject the request. BSBS follows this scheduling strategy. Algorithm 4 shows the pseudocode of BSBS. In the worst case, its complexity is $O(T \cdot |E| + (|E| + |V|log|V|))$.

From the illustration of FSBS, we know that for $(v_s, v_d, 9Gb/s, 4s, 7s)$, we are not able to schedule bandwidth of $9Gb/s$ within $[4s, 7s]$. In this case, BSBS directly sends a reject message to the user.

Algorithm 4: BSBS (Basic Streaming Bandwidth Scheduling)

INPUT: $G = (V, E)$, a BRR (v_s, v_d, B, t_S, t_E).
OUTPUT: The best bandwidth reservation option if the given BRR can be successfully scheduled or a reject message, otherwise.

1: The same as Lines 1–7 of Algorithm 1;
2: Run modified Dijkstra's algorithm to identify the path with the shortest distance from v_s to v_d within time interval $[TS[m], TS[n]]$. Suppose that the returned path is p with available bandwidth b;
3: **if** $b \geq B$ **then**
4: Create bandwidth reservation option (p, B, t_S, t_E) and return it to the user.
5: **else**
6: Send a reject message to the user.
7: **end if**

14.5 Performance Evaluation

The OSCARS of ESnet is one of the most widely used bandwidth reservation services [2, 5, 7] in broad science communities. To mimic the real-life ESnet scenarios, we construct its topology using the data gathered from ESnet and conduct extensive simulations on this real-life network topology [12].

For a random generated BRR (v_s, v_d, B, t_S, t_E), v_s and v_d are randomly selected among the collected nodes, B is set to be a random integer between $1Gb/s$ and $8Gb/s$, t_S is randomly selected from $[0, 19s]$, and t_E is randomly selected from $(t_S, 20s]$. We run 10 sets of simulations. In the i-th set of simulations, $1 \leq i \leq 10$, there are 10 different batches of $i \times 200$ randomly generated BRRs. We implement and execute both FSBS and BSBS to process the same batch of randomly generated BRRs and measure the following two performance metrics in each simulation: (1) BRR scheduling success ratio, defined as the percentage of BRRs that have been successfully scheduled within a BRR batch, and (2) average data transfer completion time of the scheduled BRRs within a BRR batch. We plot in Figs. 14.3 and 14.4 the average performance metrics and the corresponding variances with the 95% confidence level across all the 10 BRR batches in each set of simulations.

The curves of FSBS_Best and BSBS in Fig. 14.3 represent the scheduling ratios of the BRRs that have been successfully scheduled by FSBS and BSBS, respectively. The average values of the above two ratios are 84.26% and 74.14%, and the average data transfer completion times of these two groups of BRRs are 10.41 s and 10.03 s (the curves of FSBS_Best and BSBS in Fig. 14.4), respectively.

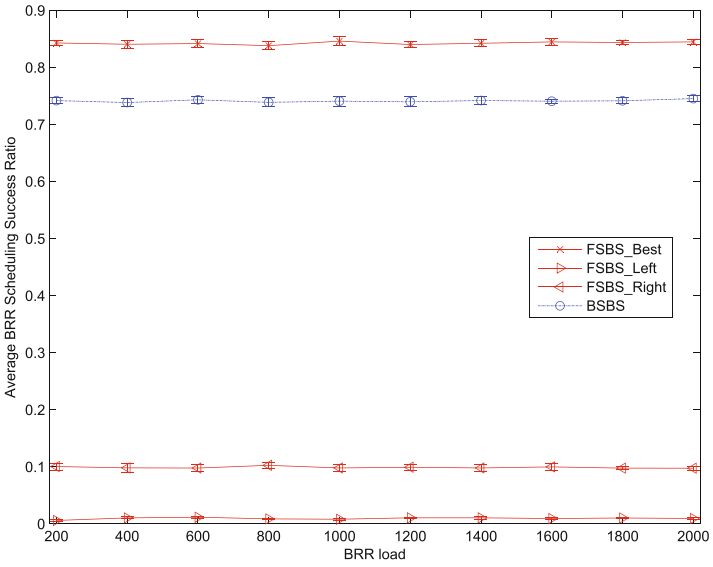

Fig. 14.3 Comparison of the BRR scheduling success ratio

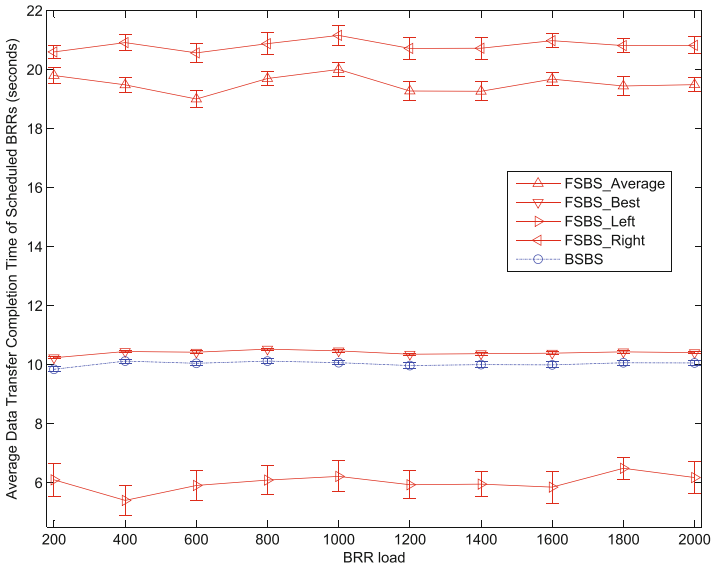

Fig. 14.4 Comparison of the average data transfer completion time of the scheduled BRRs

The curves of FSBS_Left and FSBS_Right in Fig. 14.3 represent the percentages of the BRRs within the entire batch whose closest bandwidth reservation options fall before and after the user-specified time interval, respectively. The average of the above two ratios are 0.96% and 9.90%, and the average data transfer completion times of these two groups of BRRs are 6.01 s and 20.82 s (the curves of FSBS_Left and FSBS_Right in Fig. 14.4), respectively. The average data transfer completion time of the closest bandwidth reservation options is 19.52 s as shown by the curves of FSBS_Average in Fig. 14.4.

The above performance measurements illustrate the flexibility of FSBS with an improved overall scheduling performance in comparison with BSBS.

14.6 Conclusion

In this paper, we studied flexible bandwidth scheduling for streaming data movement over dedicated networks. For a user request specifying the bandwidth reservation time interval and the desired bandwidth, if there is a lack of sufficient resources to satisfy the request, we considered providing a bandwidth reservation option to schedule the desired bandwidth within a time interval closest to the user-specified one and proposed a flexible scheduling algorithm, Flexible Streaming Bandwidth Scheduling (FSBS). For performance comparison, we also designed one basic scheduling algorithm, Basic Streaming Bandwidth Scheduling (BSBS). Extensive simulations were conducted to show the superior performance of FSBS.

The proposed scheduling algorithm has great potential to improve the stability of scientific applications running in network environments with limited resources.

It is of our future interest to develop more flexible service models and scheduling algorithms to improve network-based application performance and consider the scheduling of BRRs with different priority values in more complex environments shared by different groups of users.

References

1. Balman, M., Chaniotakisy, E., Shoshani, A., Sim, A.: A flexible reservation algorithm for advance network provisioning. In: Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–11. Washington, DC, USA (2010)
2. Charbonneau, N., Vokkarane, V.M., Guok, C., Monga, I.: Advance reservation frameworks in hybrid ip-wdm networks. IEEE Commun. Mag. **49**(5), 132–139 (2011)
3. Dart, E., Hester, M., Zurawski, J.: Fusion energy sciences network requirements review - final report 2014. In: ESnet Network Requirements Workshop (2014)
4. Guok, C., Robertson, D., Thompson, M., Lee, J., Tierney, B., Johnston, W.: Intra and interdomain circuit provisioning using the oscars reservation system. In: 3rd International Conference on Broadband Communications, Networks and Systems, pp. 1–8, San Jose, CA, USA (2006)
5. Lehman, T., Yang, X., Ghani, N., Gu, F., Guok, C., Monga, I., Tierney, B.: Multilayer networks: an architecture framework. IEEE Commun. Mag. **49**(5), 122–130 (2011)
6. Lin, Y., Wu, Q.: Complexity analysis and algorithm design for advance bandwidth scheduling in dedicated networks. IEEE/ACM Trans. Netw. **21**(1), 14–27 (2013)
7. Monga, I., Guok, C., Johnston, W.E., Tierney, B.: Hybrid networks: lessons learned and future challenges based on esnet4 experience. IEEE Commun. Mag. **49**(5), 114–121 (2011)
8. Stergiou, C., Psannis, K.E.: Recent advances delivered by mobile cloud computing and internet of things for big data applications: a survey. Int. J. Netw. Manag. **27**(3), e1930 (2016). https://doi.org/10.1002/nem.1930
9. Tanwir, S., Battestilli, L., Perros, H., Karmous-Edwards, G.: Dynamic scheduling of network resources with advance reservations in optical grids. Int. J. Netw. Manag. **18**(2), 79–105 (2008)
10. Zuo, L., Zhu, M.M.: Bandwidth provision strategies for reliable data movements in dedicated networks. In: 2016 IEEE International Conference on Big Data (Big Data), pp. 3069–3078 (2016)
11. Zuo, L., Zhu, M.M.: Improved scheduling algorithms for single-path multiple bandwidth reservation requests. In: The 10th IEEE International Conference on Big Data Science and Engineering (BigDataSE-16), pp. 1692–1699 (2016)
12. Zuo, L., Zhu, M., Wu, C.: Fast and efficient bandwidth reservation algorithms for dynamic network provisioning. J. Netw. Syst. Manag., **23**(3), 420–444 (2015)
13. Zuo, L., Zhu, M.M., Wu, C.Q.: Concurrent bandwidth scheduling for big data transfer over a dedicated channel. Int. J. Commun. Netw. Distrib. Syst. **15**(2/3), 169–190 (2015)
14. Zuo, L., Zhu, M.M., Wu, Q.Q.: Concurrent bandwidth reservation strategies for big data transfers in high-performance networks. IEEE Trans. Netw. Serv. Manage. **12**(2), 232–247 (2015)
15. Zuo, L., Zhu, M.M., Wu, C.Q., Zurawski, J.: Fault-tolerant bandwidth reservation strategies for data transfers in high-performance networks. Comput. Netw. **113**, 1–16 (2017)