Distributed Online Scheduling and Routing of Multicast-Oriented Tasks for Profit-Driven Cloud Computing

Kaiyue Wu, Ping Lu, and Zuqing Zhu, *Senior Member, IEEE*

***Abstract*—It is known that to support a few common applications well, e.g., datacenter (DC) backup, multicast-oriented tasks need to be handled in inter-DC networks. In this letter, we propose an approach to schedule and route multicast-oriented tasks in inter-DC networks with arbitrary topologies. Specifically, we leverage Lyapunov optimization to develop a distributed online approach that can maximize the time-average profit with only local information. Besides, we also design a destination grouping scheme to address the scalability issue of our proposed approach and demonstrate that the number of queues in the system can be reduced significantly. Extensive simulations verify the performance of the proposed approaches.**

***Index Terms*—Distributed online scheduling, multicast, Lyapunov optimization, datacenter, cloud computing.**

I. Introduction

OVER THE past few years, the development of cloud computing has stimulated rapid deployment of inter-datacenter (inter-DC) networks to connect geographically distributed DCs for offering high-quality and reliable services [1]. Meanwhile, we should notice that in order to support a few common applications well, *e.g.*, DC backup and migration, collaborative computing, *etc*, multicast-oriented tasks need to be handled in the inter-DC networks. Hence, it would be relevant to study how to schedule and route multicast-oriented tasks effectively in inter-DC networks.

The routing of multicast-oriented services/tasks in DC-related networks have been considered in existing work. The authors of [2] have proposed an efficient and scalable multicast routing scheme for intra-DC networks. However, the study was based on the unique topologies of intra-DC networks (*e.g.*, fat tree) and did not address task scheduling. Meanwhile, without considering task scheduling, people have investigated the multicast routing and related resource allocation schemes for optical networks in [3, 4], which can be applied to support inter-DC communications.

In this letter, we propose an approach to schedule and route multicast-oriented tasks effectively in inter-DC networks with arbitrary topologies. Specifically, we leverage Lyapunov optimization to develop a distributed online approach that can schedule and route multicast-oriented tasks based on local information to maximize the time-average profit. Note that, a similar scenario has been applied to maximize the throughput of multi-rate multicast in wireless networks in [5]. However, the authors only considered pre-determined multicast trees, which makes their proposal not suitable to the dynamic scenario addressed in this work. Hence, as we will show later in Section V, applying their approach to our problem can result in sub-optimal results. Besides, we also design a destination grouping scheme to address the scalability issue of our proposed approach, and demonstrate that the number of queues in the system can be reduced significantly.

The rest of the letter is organized as follows. Section II presents the problem formulation. The proposed distributed online algorithm is described in Section III. Section IV addresses the scalability issue of our proposal by proposing the destination grouping scheme, and we evaluate the proposed algorithms with numerical simulations in Section V. Finally, Section VI summarizes the letter.

Manuscript received October 23, 2015; accepted February 2, 2016. Date of publication February 4, 2016; date of current version April 7, 2016. This work was supported in part by the NSFC Project under Grant 61371117, in part by the Fundamental Research Funds for Central Universities under Grant WK2100060010, in part by Natural Science Research Project for Universities in Anhui under Grant KJ2014ZD38, and in part by the Strategic Priority Research Program of the CAS under Grant XDA06011202. The associate editor coordinating the review of this paper and approving it for publication was A. Mourad.

The authors are with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230027, China (e-mail: zqzhu@ieee.org).

Digital Object Identifier 10.1109/LCOMM.2016.2526001

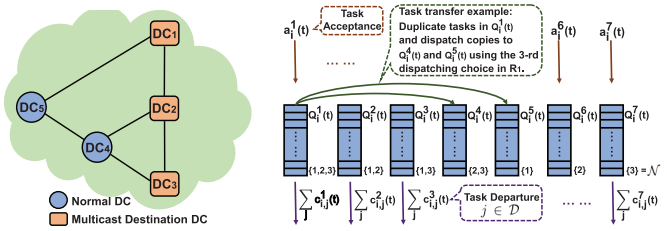
II. Problem Formulation

A. System Model

We denote an inter-DC network as $\mathcal{G}(\mathcal{D}, \mathcal{E})$, where $\mathcal{D} = \{1, \ldots, |\mathcal{D}|\}$ is the set of DCs that are interconnected by the links in set $\mathcal{E}$, as shown in Fig. 1(a). The available bandwidth on a link $(i, j) \in \mathcal{E}$, which is between two adjacent DCs i and j ($i, j \in \mathcal{D}$), is defined as $b_{i,j}$. In order to optimize the multicast schemes in the network dynamically, we assume that the network is a discrete-time system that operates on time-slots, *i.e.*, the service provisioning scheme in it can be changed at $t = \Delta t,\ 2\Delta t,\ \ldots$, which can be further normalized as $t \in \{1,\ 2,\ \ldots\}$. A multicast-oriented task needs to be delivered to multiple destination DCs (*i.e.*, denoted with set $\mathcal{N}$) for processing. Without loss of generality, we assume that in the inter-DC network, the first $|\mathcal{M}|$ DCs can be used as multicast destinations. Hence, the set of possible destination DCs is $\mathcal{M} = \{1, \ldots, |\mathcal{M}|\}$, where we have $\mathcal{N} \subseteq \mathcal{M} \subseteq \mathcal{D}$.

In each DC $i \in \mathcal{D}$, we allocate $2^{|\mathcal{M}|} - 1$ queues to buffer all the accepted tasks and categorize them based on their destination sets [6]. Then, the queue in DC i for the tasks whose destination sets equal $\mathcal{N}$ can be defined as $Q_i^{\mathcal{N}}(t)$. For instance, if we assume that $|\mathcal{M}| = 3$ for the inter-DC network in Fig. 1(a), then we need to allocate 7 queues in DC i for the multicast-oriented tasks, which are

$$\mathbf{Q}(t) = \{Q_i^{\{1,2,3\}}(t), Q_i^{\{1,2\}}(t), Q_i^{\{1,3\}}(t), Q_i^{\{2,3\}}(t), Q_i^{\{1\}}(t), Q_i^{\{2\}}(t), Q_i^{\{3\}}(t)\}, \quad \forall i \in \mathcal{D}.$$



(a) DC network. (b) Queue system in DC i.

Fig. 1. Scheduling and routing multicast-oriented tasks in inter-DC networks.

Note that, it is also necessary to sort these queues and assign a unique index to each of them. Fig. 1(b) illustrates the scheme to sort the queues and assign indices to them. Basically, the index of a queue ranges from 1 to $2^{|\mathcal{M}|}-1$, and thus we can refer to a queue $Q_i^{\mathcal{N}}(t)$ as $Q_i^k(t)$, where $k \in \mathcal{K} = \{1, \ldots, 2^{|\mathcal{M}|}-1\}$ is its unique index.

At time t, there are $A_i^k(t)$ tasks arriving at the k-th queue in DC i, and the maximum value of $A_i^k(t)$ is A^{max}, which is the upper bound on incoming tasks per unit time. $a_i^k(t)$ denotes the number of accepted tasks to the queue and satisfies

$$0 \le a_i^k(t) \le A_i^k(t), \quad \forall i \in \mathcal{D},\ k \in \mathcal{K}. \tag{1}$$

The time-average expectation of $a_i^k(t)$ is a_i^k. We introduce $c_{i,j}^k(t)$ to represent the number of tasks in k-th queue in DC i, which will be transferred to DC j. We assume that the data-size of each task is identical[1], as s. Hence, the resulting bandwidth usage on link (i, j) is $\sum_k c_{i,j}^k(t) \cdot s$, which should not exceed the available bandwidth on link (i, j), namely

$$\sum_k c_{i,j}^k(t) \cdot s \le b_{i,j}, \quad \forall i, j \in \mathcal{D},\ \forall (i,j) \in \mathcal{E}. \tag{2}$$

If we define the time-average expectation of $c_{i,j}^k(t)$ as $c_{i,j}^k$, then the total bandwidth usage on link (i, j) on average is

$$C_{i,j} = \sum_k c_{i,j}^k \cdot s, \quad \forall i, j \in \mathcal{D},\ \forall (i,j) \in \mathcal{E}. \tag{3}$$

In order to realize multicast, we need to duplicate a task and dispatch the copies to its destination DCs. This can be achieved with the following mechanism. We assume that a task's destination set is $\mathcal{N}$. Then, at each time t, we try to duplicate the task and dispatch the copies to a set of queues whose destination sets do not overlap and have a union of $\mathcal{N}$ in the current DC. For example, we can make two copies of a task in queue $Q_i^{\{1,2,3\}}(t)$ and dispatch them to queues $Q_i^{\{1,3\}}(t)$ and $Q_i^{\{2\}}(t)$, and in the mean time, the original task in $Q_i^{\{1,2,3\}}(t)$ is deleted. Next, the copies are transferred to their next-hop DC(s) and enqueue in the corresponding queues. This procedure is repeated until the copies reach all their destination DCs. Note that, a task will still be sent out even after reaching one of its destination DCs, if its destination set has not been fully covered. For instance, at DC 1, a task to DCs $\{1, 2\}$ will be processed locally and converted to one targeting to DC 2 at the same time.

[1]Note that, this assumption would not affect the generality of our model, as tasks with different data-sizes can be further divided into sub-tasks according to an identical data-size.

We define a matrix to indicate all the possible task dispatching choices of a queue. For the k-th queue, its matrix is $\mathbf{R}_k$, each row of which represents a dispatching choice and is denoted as n_k, while each of its columns corresponds to a queue. Supposing there are N_k choices for the k-th queue, we can determine the size of $\mathbf{R}_k$ as $N_k \times (2^{|\mathcal{M}|}-1)$. In $\mathbf{R}_k$, the element (n_k, m) (*i.e.*, on the n_k-th row and m-th column) equals 1 if the n_k-th choice of task dispatching sends a copy to the m-th queue, otherwise it is 0. We still use the inter-DC network in Fig. 1(a) as an example, $\mathbf{R}_1$ (*i.e.*, $k = 1$) is for $\mathcal{N} = \{1, 2, 3\}$ and we have[2]

$$\mathbf{R}_1 = \begin{bmatrix} 0,1,0,0,0,0,1 \\ 0,0,1,0,0,1,0 \\ 0,0,0,1,1,0,0 \\ 0,0,0,0,1,1,1 \end{bmatrix}_{4\times 7}.$$

We then introduce a control variable $r_i^{k,n_k}(t)$ to store the number of tasks in the k-th queue in DC i, which use the n_k-th dispatching choice at time t. Its value satisfies

$$r_i^{k,n_k}(t) \le Q_i^k(t), \quad \forall i \in \mathcal{D},\ k \in \mathcal{K}. \tag{4}$$

And we can model the queue evolution over time as

$$\begin{aligned} Q_i^k(t+1) = \max\Bigg[& \max\left(Q_i^k(t) - \sum_{n_k} r_i^{k,n_k}(t), 0\right) \\ & + \sum_{m,n_m} r_i^{m,n_m}(t) \cdot \mathbf{R}_m(n_m, k) - \sum_j c_{i,j}^k(t), 0 \Bigg] \\ & + \sum_j c_{j,i}^k(t) + a_i^k(t). \end{aligned} \tag{5}$$

B. Profit-Driven Optimization

We calculate the time-average profit of the whole system as the margin between the revenue from serving the tasks and the cost due to the bandwidth usage, which is formulated as

$$P = \alpha \cdot \sum_{i,k} a_i^k - \sum_{i,j} \beta_{i,j} \cdot C_{i,j}, \tag{6}$$

where α is the fixed revenue coefficient and $\beta_{i,j}$ is the cost of unit bandwidth on link (i, j). Basically, we assume that the cloud service provider only owns the DCs, while the inter-DC connections are rented from one or more Internet service providers. Note that, this is also the case in many multi-DC cloud systems[3]. The profit-driven optimization then becomes

$$\begin{aligned} &\text{Maximize} \quad P, \\ &s.t. \quad \text{Eqs. (1)–(4)}, \\ &Q_i^k(t) \text{ keeps steady}, \quad \forall i \in \mathcal{D},\ k \in \mathcal{K}. \end{aligned} \tag{7}$$

[2]Here, to expedite task processing, we enforce the rule that each multicast-oriented task has to be divided into those that cover smaller destination sets at each dispatching, until it becomes a unicast-oriented one. Hence, $\{1, 2, 3\} \to \{1, 2, 3\}$ is not considered as a feasible choice in $\mathbf{R}_1$.

[3]Other than the bandwidth usage, there may be other costs incurred by serving the tasks. As long as the costs are linear terms, they can fit into the model in Eq. (6). We will consider nonlinear cost terms in our future work.

III. Distributed Online Scheduling and Routing

To maximize the profit while keeping the network stable, we design a distributed online scheduling and routing algorithm. With the Lyapunov optimization techniques in [7], we transform the original optimization problem into the following three independent sub-problems. Here, due to the page limit, we omit the detailed derivations. Note that, there is apparently a tradeoff between the profit and the queue lengths in the DCs, and hence we introduce an adjustable parameter V in the following analysis to adjust this tradeoff.

1) *Acceptance Control:*

$$\begin{aligned} &\text{Minimize} \quad (Q_i^k(t) - V \cdot \alpha) \cdot a_i^k(t), \\ &s.t. \quad \text{Eq. (1).} \end{aligned} \tag{8}$$

We can get the optimal value of $a_i^k(t)$ as

$$a_i^k(t)^* = \begin{cases} A_i^k(t), & Q_i^k(t) \le V \cdot \alpha, \\ 0, & \text{otherwise}, \end{cases} \tag{9}$$

which means that the k-th queue in DC i should accept $a_i^k(t)^*$ tasks at time t.

2) *Task Transfer:* The number of tasks to be dispatched among the queues in each DC can be decided by solving the following optimization problem

$$\begin{aligned} &\text{Maximize} \quad \sum_{n_k} r_i^{k,n_k}(t) \cdot \left(Q_i^k(t) - \sum_m Q_i^m(t) \cdot \mathbf{R}_k(n_k, m) \right), \\ &s.t. \quad \text{Eq. (4).} \end{aligned} \tag{10}$$

In the n_k-th row of $\mathbf{R}_k$, we record all the non-zero elements' column numbers in set M_{n_k}. Then, Eq. (10) becomes

$$\begin{aligned} &\text{Maximize} \quad \sum_{n_k} r_i^{k,n_k}(t) \cdot \left(Q_i^k(t) - \sum_{m \in M_{n_k}} Q_i^m(t) \right), \\ &s.t. \quad \text{Eq. (4).} \end{aligned} \tag{11}$$

If we assume

$$n_k^* = \text{argmax} \left\{ Q_i^k(t) - \sum_{m \in M_{n_k}} Q_i^m(t) | \forall n_k \in [1, N_k] \right\}, \tag{12}$$

we obtain the optimal value of $r_i^{k,n_k}(t)$ as

$$r_i^{k,n_k}(t)^* = \begin{cases} Q_i^k(t), & n_k = n_k^* \wedge \sum_{m \in M_{n_k}} Q_i^m(t) < Q_i^k(t), \\ 0, & \text{otherwise}, \end{cases} \tag{13}$$

which indicates that at time t, there should be $r_i^{k,n_k}(t)^*$ tasks dispatched from the k-th queue to other queues in DC i, using the n_k-th dispatching choice.

3) *Bandwidth Allocation*: The task routing among the DCs is determined by solving the following optimization problem

$$\begin{aligned} &\text{Maximize} \quad \sum_{i,j,k} c_{i,j}^k(t) \cdot (Q_i^k(t) - Q_j^k(t) - V \cdot \beta_{i,j} \cdot s), \\ &s.t. \quad \text{Eq. (2)}, \ \sum_j c_{i,j}^k(t) \le Q_i^k(t), \ \forall i, \ (i,j) \in \mathcal{E}, \ k. \end{aligned} \tag{14}$$

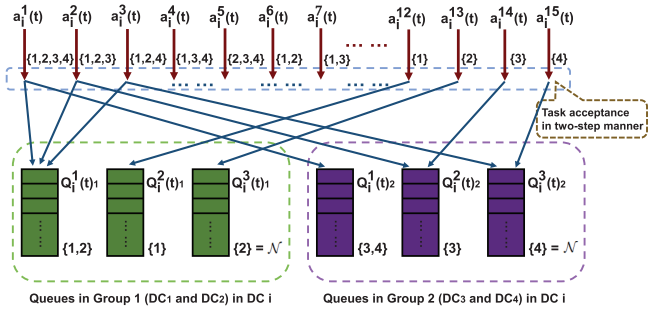


Fig. 2. Example on destination grouping.

Here, we define

$$\begin{aligned} &k^* = \text{argmax}\{Q_i^k(t) - Q_j^k(t) - V \cdot \beta_{i,j} \cdot s | \forall k \in \mathcal{K}\}, \\ &\widehat{Q_{i,j}^k(t)} = Q_i^k(t) - Q_j^k(t) - V \cdot \beta_{i,j} \cdot s, \end{aligned} \tag{15}$$

and get the optimal value of $c_{i,j}^k(t)^*$ as

$$c_{i,j}^k(t)^* = \begin{cases} \min\left(Q_i^k(t), \lfloor \frac{b_{i,j}}{s} \rfloor \right), & k = k^* \wedge \widehat{Q_{i,j}^k(t)} > 0, \\ 0, & \text{otherwise}. \end{cases} \tag{16}$$

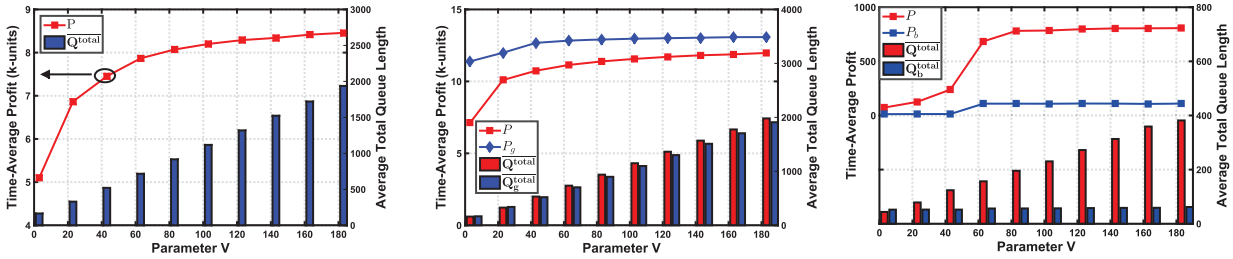
Hence, at time t, there should be $c_{i,j}^k(t)^*$ tasks from the k-th queue in DC i traveling to DC j through link (i, j).

IV. Destination Grouping for Better Scalability

With the approach discussed above, the number of queues in each DC is $2^{|\mathcal{M}|} - 1$, which increases exponentially with the size of the destination DC set $\mathcal{M}$. This makes the approach not scale well with $|\mathcal{M}|$. We solve this issue by introducing a destination grouping scheme. Specifically, we divide all the destination DCs in $\mathcal{M}$ into groups each of which has at most X members, where X is a relatively small number. Then, for the destinations in each group, we allocate queues to cover their combinations with the scheme in Section II-A. Hence, the number of queues in each DC is upper-bounded by $(2^X - 1) \cdot \lceil \frac{|\mathcal{M}|}{X} \rceil$, which only increases linearly with $|\mathcal{M}|$. The procedure proposed in Section III can still be used, with the only exception that the task acceptance will be handled in a two-step manner. Specifically, we first find the groups to cover a task's destination set, and then select specific queue(s) in each group to accept the task. Fig. 2 shows an example of the destination grouping scheme, where $|\mathcal{M}| = 4$ and $X = 2$. We can see that the number of queues decreases from 15 to 6, while all the destination combinations can still be covered.

V. Performance Evaluation

We use simulations to evaluate the performance of the proposed approach, and leverage the work in [5] to design a benchmark algorithm, *i.e.*, Lyapunov optimization with predetermined multicast trees, for performance comparison. Note that, the work in [5] tried to maximize the throughput, which is different from our optimization objective. However, since in our system, the volume of accepted tasks per unit time is just the task processing throughput, there is a positive correlation between profit and throughput, according to Eq. (6).



(a) Impact of parameter V (b) Scalability evaluation. (c) Comparisons with benchmark.

Fig. 3. Simulation results.

A. Impact of Adjustable Parameter V

We first study the impact of parameter V on the tradeoff between the time-average profit P and the average total length of all the queues in the network $\overline{Q^{total}}$. We set the maximum task arrival rate as $A^{max} = 300$, the data-size of each task is $s = 1$, and the destinations of each task are randomly chosen. The simulations use a 15-node random topology with $|\mathcal{M}| = 3$. The topology has $b_{i,j} = 60$ for all the links in $\mathcal{E}$, the link cost $\beta_{i,j}$ is randomly chosen within $(0, 10]$ for each link, and the revenue coefficient is set as $\alpha = 10$. Each simulation runs for a period of 10000 time slots and the results are obtained by averaging the outputs of 5 independent simulations. Fig. 3(a) shows the results of P and $\overline{Q^{total}}$. We can see that $\overline{Q^{total}}$ increases almost linearly with V, while P gradually approaches to a plateau when V increases. These results verify that the proposed approach works as expected.

B. Scalability Evaluation

We then evaluate the scalability of our proposed approach. Firstly, we increase $|\mathcal{M}|$ to 4 but keep all the other parameters unchanged. Here, we compare the scheme with the destination grouping with the one without it. The grouping scheme uses $X = 2$ DCs and thus obtains two groups. Fig. 3(b) plots the results of the time-average profit and the average total length of all the queues in the network, where P_g and $\overline{Q_g^{total}}$ are from the scheme with destination grouping. We observe that $\overline{Q^{total}}$ and $\overline{Q_g^{total}}$ are similar, while P_g is consistently higher than P. These observations are really promising, since they indicate that the destination grouping scheme not only deals with the scalability issue well but also makes the system more profitable. This is basically because since the destination grouping scales down the queuing system by managing the queues in different groups independently, the possibility of transmitting tasks unwisely among queues decreases and so does the cost.

The simulations then use $|\mathcal{M}| = 9$ and $X = 3$ to get three groups. Table I shows the results, which still indicate that P_g gradually approaches to a plateau when V increases. The average total lengths of the queues in the three groups, *i.e.*, $\overline{Q_1}$, $\overline{Q_2}$ and $\overline{Q_3}$, exhibit similar values at each V, and thus the traffic loads in the groups are balanced well.

C. Performance Comparisons With Benchmark

Finally, we evaluate our proposed approach against the benchmark developed based on the work in [5]. For the benchmark, all the multicast trees are properly chosen to optimize its performance, and to address the complexity of the multicast tree calculation, we use the 5-node topology in Fig. 1(a) and set $|\mathcal{M}| = 3$ and $A^{max} = 500$. As the inter-DC network is small, our approach does not group destinations. Fig. 3(c) shows the results of the time-average profit and average total length of all the queues, where P_b and $\overline{Q_b^{total}}$ are from the benchmark. Our approach is more profitable than the benchmark, and when V increases, P approaches to a much higher plateau than P_b. The tradeoff is that $\overline{Q^{total}}$ is longer than $\overline{Q_b^{total}}$, which is because the benchmark always uses pre-determined multicast trees with optimized paths, the queuing overheads are reduced.

TABLE I
RESULTS OF DESTINATION GROUPING SCHEME ($|\mathcal{M}| = 9$, $X = 3$)

V	P_g	$\overline{Q_1}$	$\overline{Q_2}$	$\overline{Q_3}$
1	288301.0	6849.79	6545.79	7084.97
5	289058.5	9548.54	8927.34	9372.31
9	289300.3	9608.15	9638.30	9566.10

VI. Conclusion

We proposed a distributed online approach to schedule and route multicast-oriented tasks in inter-DC networks for maximizing the time-average profit. A destination grouping scheme was also designed to address the scalability issue of the proposal. Simulation results indicated that our proposal could outperform the existing approach in terms of profit.

References

[1] P. Lu *et al.*, "Highly-efficient data migration and backup for big data applications in elastic optical inter-datacenter networks," *IEEE Netw.*, vol. 29, no. 5, pp. 36–42, Sep./Oct. 2015.
[2] D. Li *et al.*, "ESM: Efficient and scalable data center multicast routing," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 944–955, Jun. 2012.
[3] L. Yang *et al.*, "Leveraging light-forest with rateless network coding to design efficient all-optical multicast schemes for elastic optical networks," *J. Lightw. Technol.*, vol. 33, no. 18, pp. 3945–3955, Sep. 2015.
[4] S. Li, W. Lu, X. Liu, and Z. Zhu, "Fragmentation-aware service provisioning for advance reservation multicast in SD-EONs," *Opt. Express*, vol. 23, pp. 25804–25813, Oct. 2015.
[5] G. Paschos *et al.*, "Multirate multicast: Optimal algorithms and implementation," in *Proc. INFOCOM*, Apr. 2014, pp. 343–351.
[6] M. Marsan *et al.*, "Multicast traffic in input-queued switches: Optimal scheduling and maximum throughput," *IEEE/ACM Trans. Netw.*, vol. 11, no. 3, pp. 465–477, Jun. 2003.
[7] M. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan and Claypool, 2010.