# Manycasting Over
# Optical Burst-Switched Networks

Xiaodong Huang[†], Qingya She[†], Vinod M. Vokkarane[‡], and Jason P. Jue[†]
[†]Department of Computer Science
The University of Texas at Dallas, Richardson, TX 75083
[‡]Department of Computer and Information Science
University of Massachusetts, Dartmouth, MA 02747
Email: {xxh020100, qxs032000, jjue}@utdallas.edu, vvokkarane@umassd.edu

***Abstract*— In this paper, we discuss for the first time the issue of supporting manycasting service over optical burst-switched (OBS) networks. One of the primary challenges in providing manycasting service over OBS networks is to reduce data loss due to burst contentions. We propose two new schemes, *static over-provisioning* (SOP) and *dynamic membership* (DM), to alleviate this data loss problem. The proposed schemes take into consideration the specific properties of manycasting, and the schemes may complement existing contention resolution schemes. The effectiveness of the proposed schemes is verified through simulation.**

**Keywords:** Multicast, Manycast, IP, WDM, and OBS.

## I. INTRODUCTION

The *manycast problem*, also referred to as the quorumcast problem and the k-Steiner tree problem, was first proposed in 1994 independently by [1] and [2], and is defined as follows: given a network $G(V, E)$, an edge cost function $g : E \rightarrow R^+$, an integer $k$, a source $s$, and a subset of candidate destinations $D_c \subseteq V$, $|D_c| = m \geq k$, find a minimum cost tree spanning $k$ destinations in $D_c$. The cost of a tree is defined as a sum of the cost of edges on the tree. A manycast request can simply be denoted as $(s, D_c, k)$. A subtle difference between manycast and multicast is that, in manycast, the *actual* destinations to be covered are to be determined instead of being given as in multicast. The manycast problem is NP-hard [2].

To support manycasting service in IP over optical burst-switched (OBS) networks, we first need to decide which layer should be responsible for selecting $k$ out of $m$ candidate destinations. If the selection is done at the IP layer, manycast requests become multicast requests to the OBS layer, and it is sufficient that OBS networks support only multicasting service. However, for an overlay network architecture, which is the most used network architecture in practice, the IP layer usually does not have much information about the OBS layer. Then the selection of $k$ destinations by the IP layer is similar to the *random algorithm* in [1], which has been proved to have poor performance. Therefore, supporting manycasting at the OBS layer is necessary for bandwidth-efficient manycasting.

The manycast problem has already attracted much research attention [1]-[6]. These work focus on finding good routing algorithms to minimize the total cost for a manycast request. To date, the best available routing algorithm for the manycast problem in a general network is a 4-approximation algorithm proposed by [6]. In this paper, we are not proposing a better routing algorithm for the manycast problem. Instead, our focus is on the data loss issue of manycasting service over OBS networks.

We first give a brief introduction to the OBS network. Optical burst switching is a promising technique for future wavelength division multiplexing (WDM) networks which are capable of providing up to terabit/s bandwidth [7] [8]. A typical OBS network works as follows. Multiple packets to the same egress edge node are packed together into a *data burst* at ingress edge nodes. The control information for a data burst, contained in a *burst header packet (BHP)*, is transmitted on a separate control channel. BHPs are processed electronically at each intermediate node to reserve network resources before the data burst arrives at a node. Data bursts will then be routed all-optically on data channels through the network.

Data loss due to *burst contention* is a special issue in OBS networks, because of the burstiness of IP traffic and the lack of effective optical buffering technique. Burst contention occurs when multiple bursts contend for the same outgoing channel on the same wavelength at the same time. There are many solutions to reduce the impact of burst contentions, such as in [9]-[12]. In this paper, we propose two new schemes, *static over-provisioning* (SOP) and *dynamic membership* (DM), to alleviate the data loss problem in manycasting. The proposed schemes take into consideration the specific properties of manycast. The proposed schemes are not a replacement of existing contention resolution schemes but a complement to those schemes. That is, the proposed schemes could be used together with existing contention resolution schemes to further reduce data loss due to burst contentions.

The rest of this paper is organized as follows. In Section II, we discuss issues of supporting manycasting service over OBS networks. In Section III, we elaborate the details of the two new schemes to reduce data loss for manycasting service. Simulation results are presented in Section IV. The paper is concluded with Section V.

## II. MANYCASTING SERVICE

The manycasting service over OBS networks is a service which will send a manycast burst $(s, D_c, k)$ from the source $s$ to $k$ destinations among candidate destinations in $D_c$. Due to burst contention, this service is a best-effort service and there is no guarantee that there will be exactly $k$ destinations that actually receive the burst. In the rest of this section, we discuss the details of implementing such a manycasting service over OBS networks. Since manycasting is a generalization of multicasting, the general ideas of multicasting over OBS networks [13] [14] [15] are largely applicable here for manycasting. The difference from multicasting is that, for manycasting, we need an independent or a coordinated process to choose $k$ destinations, in addition to the routing to these destinations.

We first discuss the routing issue. From a hardware point of view, optical multicasting is a fundamental function to support manycasting over optical networks. Optical multicasting could be implemented by multicast capable optical cross connects (MC-OXCs) using optical splitters [16] [17]. Through MC-OXCs, optical route trees could be set up dynamically for manycasting requests. The next question is how to calculate the route tree. As we mentioned before, the best routing algorithm is a 4-approximation algorithm with polynomial complexity [6]. However, this algorithm is still too complex for OBS networks with dynamic traffic. Note that, in IP multicasting, shortest-path tree (SPT) [18] is the only routing algorithm used for source specific IP multicasting, due to ease of implementation and the efficiency of computation [19]. For the same reasons, we propose a variation of the shortest-path tree algorithm for manycasting as follows.

- *Step 1*: Find the shortest paths from the source $s$ to all destinations in $D_c$.
- *Step 2*: All destinations in $D_c$ are sorted into a non-decreasing order according to the shortest distances from the source $s$ to the destinations.
- *Step 3*: The first $k$ destinations are selected and then the shortest paths from the source to these destinations are merged to obtain the route tree for the manycast request.

In terms of the network size $n$, the three steps are with time complexity $O(n^2)$, $O(1)$, and $O(n)$, respectively. If the shortest paths from the source to all destinations are known, the above SPT algorithm has linear time complexity $O(n)$. After the route tree is calculated, a control message may be sent out along the route tree to set up the multicast routing tables at involved nodes. When a manycast burst arrives at a node, the multicast table is consulted to find next hop(s) for the burst. A new BHP will be generated for each next hop and sent to the next hop.

The SPT algorithm can also be implemented in a distributed manner. *Step 1* is implemented by a unicast routing protocol, which results in a unicast routing table at each node. *Step 2* is executed at the source node in constant time. *Step 3* works as follows. After *Step 2*, the source will embed the list of selected $k$ destinations into the BHP packet. Upon receiving a BHP, the node looks up its unicast routing table to find the next hop for each destination listed in the header. For each next hop, a new BHP is generated which includes only those destinations that should be routed through this next hop node. This process continues until the packet reaches the destinations or is discarded at some intermediate node. The complexity of the distributed version of the SPT algorithm is reduced to constant time $O(1)$ at each involved node.

Bursts for manycast could be assembled in the same way as bursts for unicast. When a burst is ready to transmit, a BHP will be sent out along the route for the manycast request. The well-known OBS signaling protocols for unicast traffic, i.e., tell-and-wait (TAW), tell-and-go (TAG), just-in-time (JIT) and just-enough-time (JET) [20], can be used for manycasting with the modifications described in the above centralized or distributed version of the SPT algorithm. In the source and intermediate nodes, we adopt a *partial blocking policy*, in which, if a burst is blocked at any child branch, the other copies of the burst will continue to be routed to those unblocked child branches. The rationale behind this policy is that an OBS network will do its best to reach as many destinations as possible, and then, if necessary, a higher layer will retransmit data to those blocked destinations. Thus, bursts can be scheduled independently for each child branch by using classic scheduling algorithms, such as LAUC and LAUC-VF [21]. Note that, manycasting service could also be implemented by using other blocking policies and scheduling algorithms.

## III. NEW SCHEMES TO REDUCE DATA LOSS

Due to the lack of effective optical buffers in OBS networks and the one-way resource reservation mechanism, data loss resulting from burst contention cannot be avoided in typical OBS networks. In this section, we propose two new schemes, *static over-provisioning* (SOP) and *dynamic membership* (DM), to alleviate the data loss problem in manycasting. The proposed schemes take into consideration the specific properties of manycast. The idea behind the new schemes is to improve data availability through controlled redundancy.

The first scheme, *static over-provisioning* (SOP) is motivated by the following observation. Because of the loss property of OBS networks, even at low load, we cannot guarantee that a burst will be received by $k$ destinations, even if we could find the optimal solution (optimal $k$ destinations out of $D_c$ and the optimal route for the $k$ destinations) for a manycast request $(s, D_c, k)$. Instead of trying to avoid or resolve burst contentions such that a burst could reach the designated destination(s), we send the burst to a total of $k+k'$ destinations instead of only $k$ destinations as indicated by the request, where $0 \leq k' \leq |D_c|-k$. As a special case, if $k'=0$, there is no over-provisioning. If the bursts to some of the $k+k'$ destinations are lost, the total number of destinations which actually receive the burst may still be $k$ or more with a high probability, such that the user requirement of a manycast request is satisfied. We then need to study two sub-problems: 1) how to decide the number $k'$, given the request $(s, D_c, k)$ and the network status; and 2) how to choose these additional $k'$ destinations. In this paper, we focus on the second sub-problem and evaluate the impact of a given $k'$ on the network performance using SOP.

Here are the details of the SOP scheme. In SOP, the additional $k'$ destinations will be decided before the burst (actually the BHP of the burst) is sent out from the source node. SOP could be used with either the centralized or the distributed version of the SPT routing algorithm, with a simple extension as follows: before the execution of *Step 3* in the SPT algorithm, we increase the value of $k$ to the value of $k+k'$. By this extension, $k'$ additional destinations, which have the shortest distances after $k$ destinations have been selected, are included into the route tree. This extension is consistent with the idea of choosing the first $k$ destinations which have the shortest distance from the source. Although we choose the additional $k'$ destinations in this manner, alternatives are possible, such as choosing the additional $k'$ destinations that have the least overlap with the route tree for the first $k$ destinations.

The second scheme, *dynamic membership* (DM), takes a different perspective from SOP. In the SPT algorithm of Section II and the SOP scheme, the designated destinations and the route tree are decided at the source node, which is independent of network status. After that, the manycast request actually becomes a multicast request, which will be routed along the pre-calculated route tree. Since it is difficult for the source node to obtain the exact status information of nodes along the route tree, it may be a better choice to postpone such a decision until the burst actually arrives at intermediate nodes. In order to obtain such a flexibility, destination information should be included in the BHPs. Thus, the proposed DM scheme will work well with the distributed version of the SPT algorithm.

Here are the details of the DM scheme. In DM, a designated set of $k$ destinations is tentatively set up at the source node as before. Instead of discarding the remaining $(|D_c| - k)$ destinations, we evenly distribute the remaining destinations into all child branches at the source node. With the extra destinations, each burst that arrives at an intermediate node is still a manycast burst instead of a multicast burst. Then, if any designated destination is blocked at an intermediate node, we may send the burst to some of these extra destinations such that the total number of destinations which actually receive the burst is still no less than $k$. Therefore, in DM, the designated set of $k$ destinations may change dynamically along the route tree according to the status of the network. In turn, the route tree itself may change accordingly. The algorithm for DM with the distributed version of SPT works as follows:

- [Input]: a manycast $(u, D_u, k_u)$ arrives at Node $u$ (the source or an intermediate node) with a candidate destination set $D_u$, among which $k_u$ destinations are expected to be chosen as the actual destinations.
- [Output]: a list of $(v_i, D_{v_i}, k_{v_i})$ manycast requests to the next hop Node $v_i$, $i = 1, 2, \cdots, z$, where $z$ is the number of child branches.
- *Step 1*: If $u \in D_u$, a copy of the burst will be dropped locally at Node $u$ and update $D_u \leftarrow D_u - u$, $k_u \leftarrow k_u - 1$.
- *Step 2*: Sort the destinations in $D_u$ in non-decreasing order according to the shortest distance from Node $u$ to each destination.
- *Step 3*: Sequentially handle the destinations one by one from the ordered list until $k_u$ destinations are successfully scheduled or all destinations are processed. For each destination $d_i \in D_u$, we find the next hop Node $v_i$ to the destination from the unicast routing table. If Link $\langle u, v_i \rangle$ is freely available for the burst, $D_{v_i} \leftarrow D_{v_i} + d_i$, $k_{v_i} \leftarrow k_{v_i} + 1$. Otherwise, destination $d_i$ is discarded.
- *Step 4*: For those untouched candidate destinations in *Step 3*, sequentially assign these nodes one by one $d_i$ to the list of next hops Node $v_i$ in a Round-Robin fashion, i.e., $D_{v_i} \leftarrow D_{v_i} + d_i$. Then, the algorithm terminates.

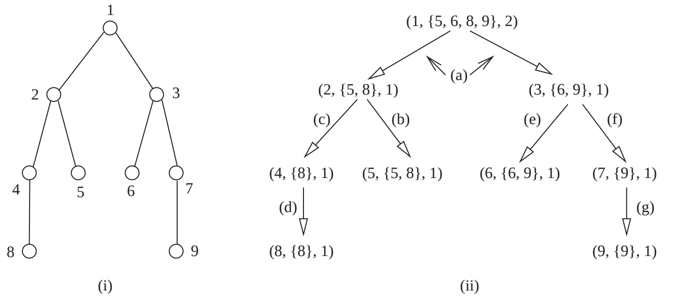

Fig. 1. An illustrative example for the DM scheme with distributed SPT. (i) a shortest-path routing tree in a network; (ii) the routing of a manycast $(1, \{5, 6, 8, 9\}, 2)$ from source Node 1. We consider the following cases: (a) Link $\langle 1, 2 \rangle$ and Link $\langle 1, 3 \rangle$ are free; (b) Link $\langle 2, 5 \rangle$ is free; (c) Link $\langle 2, 5 \rangle$ is blocked and Link $\langle 2, 4 \rangle$ is free; (d) Link $\langle 4, 8 \rangle$ is free; (e) Link $\langle 3, 6 \rangle$ is free; (f) Link $\langle 3, 6 \rangle$ is blocked and Link $\langle 3, 7 \rangle$ is free; (g) Link $\langle 7, 9 \rangle$ is free.

At the source node, the input to the above algorithm will be the manycast request itself $(s, D_c, k)$. In *Step 4*, by distributing the untouched destinations evenly among the child branches, we may expect that each branch obtains some redundant protection from potential destination blocking. Note that DM is different from the well-known deflection routing scheme. In DM, if a destination is blocked, the destination is discarded and an alternative will be chosen to replace the blocked one. On the contrary, in deflection routing, if a destination is blocked on its primary route, an alternative route (if available) will be used to route the burst to the same destination.

Let us give an example to illustrate how the proposed schemes work. Fig. 1 (i) depicts a shortest-path routing tree rooted at Node 1 and covering Node 2 through Node 9. Other links between nodes are not shown for clarity. A manycast request $(1, \{5, 6, 8, 9\}, 2)$ originates from Node 1 and intends to reach two destinations out of the candidate destinations $\{5, 6, 8, 9\}$. With regular SPT routing, we reduce the request to $(1, \{5, 6\}, 2)$. With SOP, we may instead reduce the request to $(1, \{5, 6, 8\}, 3)$, assuming $k' = 1$. Suppose the burst to Node 5 is lost at Link $\langle 2, 5 \rangle$. With regular SPT, the burst reaches only a single destination $\{6\}$. On the contrary, with SOP, the burst may still reach two destinations $\{6, 8\}$ such that the request is fulfilled. For DM, we consider the case in which both links $\langle 1, 2 \rangle$ and $\langle 1, 3 \rangle$ are free, as indicated in Fig. 1 (ii) (a). In this case, the scheduling at Node 1 to Node 5 and Node 6 succeeds. The remaining destinations, Node 8 and Node 9, will then evenly be dispatched into one of the two branches. We obtain two manycasts $(2, \{5, 8\}, 1)$ and $(3, \{6, 9\}, 1)$. At Node 2, if Link $\langle 2, 5 \rangle$ is free, the burst will be only sent to

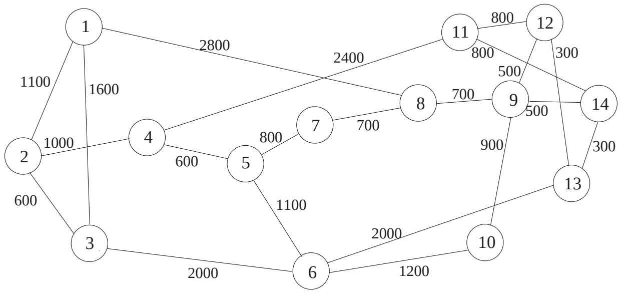

Fig. 2. The NSF network with 14 nodes and 21 bi-directional links (link distance in kilometers).

Node $5$, since the distance from Node $2$ to Node $5$ is shorter than that to Node $8$, as the case in Fig. 1 (ii) (b); if Link $\langle 2, 5\rangle$ is blocked and Link $\langle 2, 4\rangle$ is free, Node $5$ is discarded from the set of candidate destinations and the manycast will be sent to Node $8$ through Node $4$, which is shown in (c) and (d) of Fig. 1 (ii). The handling of the other branch from Node $1$ to Node $3$ is similar and is shown in (e)-(h) of Fig. 1 (ii). As we can see from this example, the proposed new schemes may reduce the blocking probability of manycasting service.

With either SOP or DM, there is no guarantee that consecutive burst transmissions for the same manycast session will reach exactly the same set of destinations. We refer to this phenomena as *non-deterministic receiving* (NDR). NDR may be desirable or non-desirable, depending on the manycast application. For example, in quorum consensus, it is not required that the set of destinations which receive the data are exactly the same from transmission to transmission. Statistically, either SOP or DM can achieve some kind of load balancing. In this case, NDR is probably desirable. In another example, database protection via replication, a snapshot of the data should be sent through continual transmissions to the exact same set of sites among possible candidate sites. In this case, NDR will corrupt the integrality of the data. However, if the data can be transmitted with one transmission (such as placing it into one burst), NDR may again become desirable to achieve load balance.

## IV. SIMULATION RESULTS

In this section, we present numerical results from our simulations. We evaluate the performance in terms of *average request blocking ratio* and *average request delay*. We first give definitions of the metrics. Let $q$ be the total number of requests in the simulation. Consider a manycast request $(s, D_c^f, k)$. Let $D'_f$ be the set of destinations which actually receive the data. If there are more than $k$ destinations actually receiving the data (with SOP), it is equivalent to $k$ destinations from the user's point of view. Thus, we define the *average request blocking ratio* as $\bar{b} = \sum_f [1.0 - \min(|D'_f|, k)/k]/q$. The *average request delay* is defined as $\bar{t} = (\sum_f \sum_{i \in D'_f} t_i)/(\sum_f |D'_f|)$, where $t_i$ is the delay from the source to the destination Node $i$. Both definitions are applicable to SPT with or without SOP and DM.

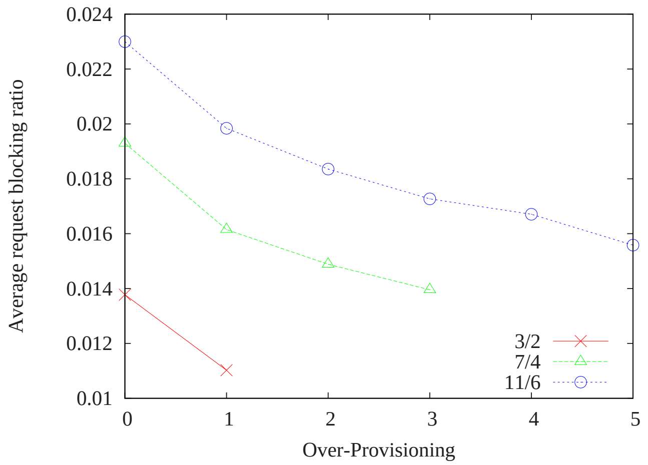

Fig. 3. The blocking performance of SOP with different levels of over-provisioning. $(0)$ means there is no over-provisioning. All others are for the SOP scheme. The network load is 0.1.

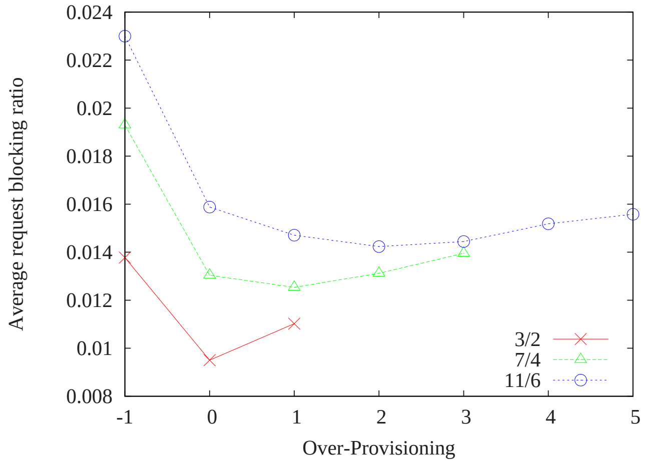

Fig. 4. The blocking performance of DM and DM plus SOP with different levels of over-provisioning. Here, for convenience, $(-1)$ denotes the case of regular SPT without DM or SOP. $(0)$ denotes the case of DM without SOP. All others are for the cases of DM plus SOP. The network load is 0.1.

Next we discuss the simulation settings. We use the NSF network in our simulation as shown in Fig. 2. All links are bidirectional. All links have the same transmission rate of $10$ Gb/s. Bursts arrive to the network according to a Poisson process with an arrival rate of $\lambda$ bursts per second. The length of a burst is exponentially distributed with expected service time of $1/\mu$ seconds. The network load is then defined as $\lambda/\mu$. The source and candidate destinations of a manycast request are evenly distributed among all nodes. There are no optical buffers or wavelength converters. As a first step, only a single wavelength plane is considered in the simulation. As in [1] [5], we consider candidate destination set $D_c$ at small, medium, and large sizes, and the intended number of destinations is a majority of the group. We use notation $m/k$ to denote a manycast request with group size $m$ and intended number of destinations $k$. To be specific, we will consider three typical

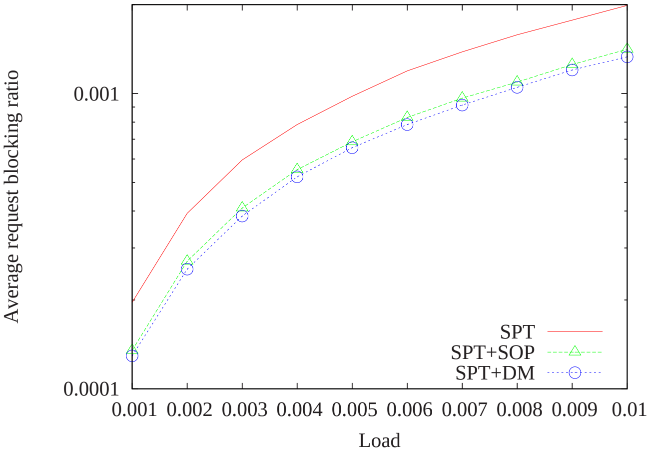

Fig. 5. The blocking performance comparison between regular SPT routing, the best configuration of SOP ($k' = 3$), and DM, for manycast configuration $7/4$ [at low load].

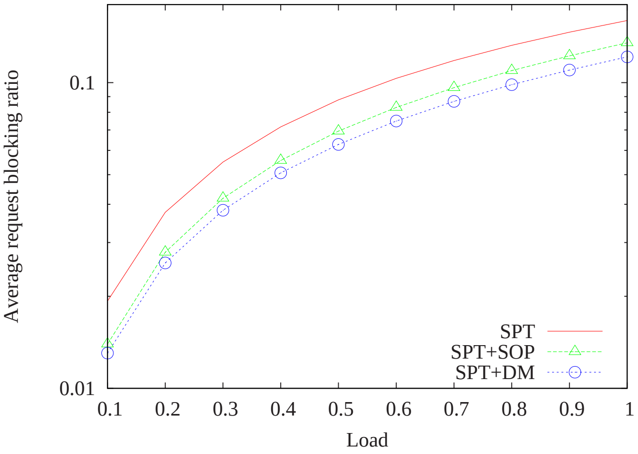

Fig. 6. The blocking performance comparison between regular SPT routing, the best configuration of SOP ($k' = 3$), and DM, for manycast configuration $7/4$ [at medium load].

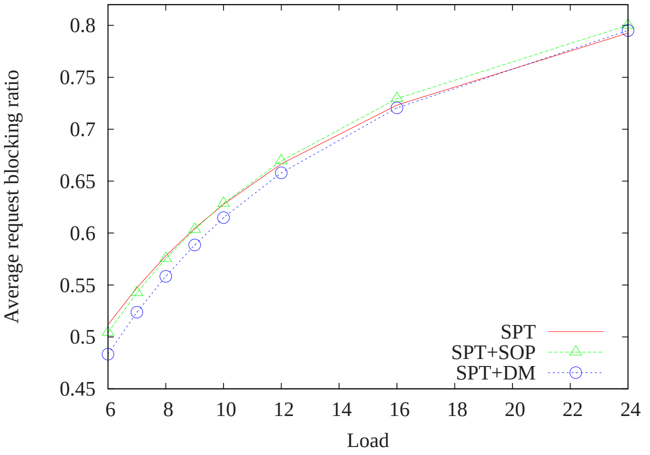

Fig. 7. The blocking performance comparison between regular SPT routing, the best configuration of SOP ($k' = 3$), and DM, for manycast configuration $7/4$ [at high load].

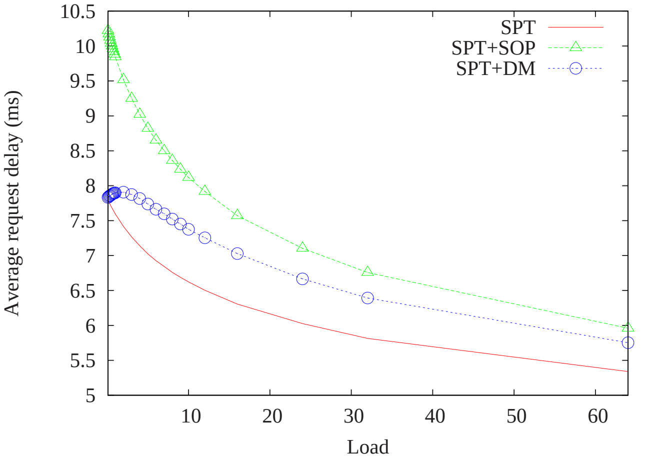

Fig. 8. The delay performance comparison between regular SPT routing, the best configuration of SOP ($k' = 3$), and DM, for manycast configuration $7/4$.

configurations: 3/2, 7/4, and 11/6.

We first evaluate the blocking performance of the SOP scheme with different levels of over-provisioning. The results are shown in Fig. 3, where the network load is 0.1. Similar results were observed under other network load values. It is clear that the SOP scheme can effectively reduce data loss. The more destinations we over-provision, the lower the average request blocking ratio. The blocking performance of DM and DM plus SOP with different levels of over-provisioning is shown in Fig. 4, where the network load is 0.1. Similar results were observed under other network load values. It can be observed that the DM scheme can significantly reduce data loss. Combined with SOP, data loss is further reduced to some extent. As opposed to SOP above, too much over-provisioning may degrade the performance of DM plus SOP. As shown in Fig. 4, the optimal over-provisioning in DM plus SOP depends on the manycast configuration $m/k$. In practice, we should use DM plus SOP with caution or may simply choose DM.

Next, we compare the performance between regular SPT routing, the SOP scheme with the best over-provisioning, and DM. We take manycast configuration $7/4$ as an example and similar results are observed for manycast $3/2$ and $11/6$. The average request blocking ratio under different ranges of network load is shown in Fig. 5 to Fig. 7. It is observed that the DM scheme has a slightly better blocking performance than the SOP scheme with the best over-provisioning (here, $k' = 3$), while both schemes can significantly improve the blocking performance over regular SPT routing when the average request blocking ratio is below 55%. The delay performance is shown in Fig. 8. The result is as expected in that the SPT algorithm has the least delay and that both schemes will slightly increase the delay. This can be explained as follows. On one hand, regular SPT algorithm chooses only the first $k$ destinations which are with the shortest distances from the source. On the other hand, both SOP and DM tend to send the burst to extra destinations which have longer distance than those nodes chosen by regular SPT algorithm. Note that, in the definition of the average request delay $\bar{t}$, we take the average of delays of all destinations receiving the burst.

## V. CONCLUSION

In this paper, we discuss the issues of supporting manycasting service over OBS networks. We propose a simple yet efficient routing algorithm for manycasting, which is based on the classic shortest-path tree algorithm. Our discussion shows that, by using multicast capable OXCs, only minor changes are needed for the well-studied OBS network architecture which aims at unicasting to support manycasting service. In this paper, we pay special attention to the data loss issue in manycasting due to burst contentions. We propose two new schemes, *static over-provisioning* (SOP) and *dynamic membership* (DM), to alleviate this data loss problem. The proposed schemes take into consideration the specific properties of manycasting. The new schemes are a complement to existing contention resolution schemes. The effectiveness of the proposed schemes is verified by our simulations.

## ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation (NSF) under grant number ANI-0133899 and CNS-0627128.

## REFERENCES

[1] S. Y. Cheung and A. Kumar, "Efficient quorumcast routing algorithms," *The Proceeding of INFOCOM, IEEE*, pp. 840-847, Toronto, Ontario, Canada, June 1994.

[2] R. Ravi, R. Sundaram, M. V. Marathe, D. J. Rosenkrantz, and S. S. Ravi, "Spanning trees short or small," *Proceedings of the fifth annual ACM-SIAM symposium on Discrete algorithms*, pp. 546-555, 1994.

[3] B. Du, J. Gu, D.H.K. Tsang, and W. Wang, "Quorumcast routing by multispace search," *Globecom96, IEEE*, vol. 2, pp. 1069 - 1073, Nov. 1996.

[4] C. P. Low, "Optimal quorumcast routing," *Globecom 98, IEEE*, vol. 5, pp. 3013 - 3016, Nov. 1998.

[5] B. Wang and J. C. Hou, "An efficient QoS routing algorithm for quorumcast communication," *Network Protocols, 2001. Ninth International Conference on, IEEE*, pp. 110-118, Nov. 2001.

[6] N. Garg, "Saving an epsilon: a 2-approximation for the k-mst problem in graphs," in *ACM Symposium on Theory of Computing*, pp. 396-402, 2005.

[7] C. Qiao and M. Yoo, "Optical burst switching (OBS) - a new paradigm for an optical Internet," *Journal of High Speed Networks*, vol. 8, no.1, pp. 69-84, Jan. 1999.

[8] J. P. Jue and V. M. Vokkarane, "Optical burst switched networks," Springer, Optical Networks Series, 2005.

[9] B. Ramamurthy and B. Mukherjee, "Wavelength conversion in WDM networking," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 7, pp. 1061-1073, Sept. 1998.

[10] C. Gauger, "Dimensioning of FDL buffers for optical burst switching nodes", *Proceedings, Optical Network Design and Modeling (ONDM 2002)*, Torino, Feb 2002.

[11] C. Hsu, T. Liu, and N. Huang, "Performance analysis of deflection routing in optical burst switched networks," *Proceedings, INFOCOM 2002, IEEE*, vol. 1, Jun. 2002.

[12] V. M. Vokkarane and J. P. Jue, "Segmentation-based non-preemptive channel scheduling algorithms for optical burst-switched networks," *IEEE/OSA Journal of Lightwave Technology (JLT)*, Special Issue on Optical Networks, vol. 23, no. 10, pp.3125-3137, Oct. 2005.

[13] C. Qiao, M. Jeong, A. Guha, X. Zhang, and J. Wei, "WDM Multicasting in IP over WDM Networks," *The proceding of IEEE International Conference on Network Protocols*, 1999.

[14] I. Baldine, G. N. Rouskas, H. G. Perros and D. Stevenson, "Signaling support for multicast and QoS within the JumpStart WDM burst switching architecture," *Optical Netowrks Magazine*, vol. 4, no. 6, pp. 68-80, Nov./Dec. 2003.

[15] X. Huang, Q. She, T. Zhang, K. Lu, and J. P. Jue, "Small group multicast in optical burst switched networks," *The proceding of, IEEE/CreateNet WOBS 2006*, co-lacated with IEEE/CreateNet BroadNets 2006, San Jose, CA, Oct. 2006.

[16] L. H. Sahasrabuddhe and B. Mukherjee, "Light trees: optical multicasting for improved performance in wavelength routed networks," *Communications Magazine, IEEE*, vol. 37, issue 2, pp. 67 - 73, Feb. 1999.

[17] W. S. Hu and Q. J. Zeng, "Multicasting optical cross connects employing splitter-and-delivery switch," *IEEE Photonics Technology Letters*, vol. 10, no. 7, pp. 970-972, Jul. 1998.

[18] F. K. Hwang, D. S. Richards, and P. Winter, "The Steiner tree problem", Elsevier Science Publishers, 1992.

[19] L. H. Sahasrabuddhe, B. Mukherjee, "Multicast routing algorithms and protocols: a tutorial," *IEEE Network*, vol.14, no.1, pp.90-102, Jan/Feb 2000.

[20] Y. Chen, C. Qiao, and X. Yu, "Optical burst switching: a new area in optical networking research," *IEEE Network*, vol. 18, no. 3, pp. 16-23, May-June 2004.

[21] Y. Xiong, M. Vandenhoute, and H. C. Cankaya, "Control architecture in optical burst-switched WDM networks", *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1838-1851, 2000.