2011

Minimizing Resource Blocking Rate in GoOBS

Shivashis Saha
University of Nebraska-Lincoln, ssaha@cse.unl.edu

Jitender S. Deogun
University of Nebraska-Lincoln, jdeogun1@unl.edu

Lisong Xu
University of Nebraska-Lincoln, xu@cse.unl.edu

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.
IEEE Global Telecommunications Conference (GLOBECOM 2011), 2011
doi: 10.1109/GLOCOM.2011.6133621

Minimizing Resource Blocking Rate in GoOBS

Shivashis Saha, Jitender S. Deogun, Lisong Xu
Department of Computer Science and Engineering,
University of Nebraska-Lincoln, Lincoln, NE 68588-0115, U.S.A.
Email: {ssaha, deogun, xu}@cse.unl.edu

***Abstract*—The state of the resources at a destination in Grid computing over OBS architecture (GoOBS) may change between a task's selection of a destination and its arrival at the destination. These changes in the availability of the resources requested at the destination may lead to blocking of tasks, and thus increase the resource blocking rate.**

In this paper, we investigate the resource scheduling problem in GoOBS. Our objective is to minimize the resource blocking rate by containing the impact of the changes in the availability of the resources at a destination. We propose a *non-selfish destination selection paradigm* to minimize the resource blocking rate. The selection of a destination by a request is called *non-selfish*, if the selected destination has sufficient resources available to simultaneously process one or more additional requests. Extensive simulations were performed to validate the effectiveness of the heuristics based on the non-selfish destination selection paradigm. Among the proposed heuristics, the NFFD heuristic is most effective in minimizing the resource blocking rate. Compared to the best existing approach, the NFFD heuristic reduces the resource blocking rate by 21% to 73% in our experiments.

I. Introduction

Next-generation grid or cloud computing applications increasingly demand high-speed, high-bandwidth networks. All-optical networks provide a good solution for such applications [1]. Grid/cloud applications are supported by a network in which each node or site represents a grid. A user submits a task or request to a site called *source*. The information about the data and the resources needed is contained in the request. These resources include but not limited to processing units, storage capacity, and software [2]. The resources requested (if available in the network) are identified for this request from one or more sites called *destinations*. The request is then transmitted from the source to the identified destination(s) using different transmission models. In anycast transmission, a destination is selected from a set of candidate destinations as compared to a fixed destination in unicast. Anycast significantly improves the performance of many grid applications as compared with unicast transmission [3]. Thus, the two important factors for such grid applications are efficient selection of destination(s), and establishing communication path between a source and the destination(s) [2], [4].

The state of resources at a destination may change between a request's selection of a destination and its arrival at the destination [2], [4]. Thus, when a request arrives at a destination the resources required for execution may no longer be available resulting in collision of requests at a destination. This *collision at destinations* may lead to higher resource blocking rate and consequently to poor performance of grid applications over Optical Burst Switching (OBS) architecture [2], [4].

In this paper, we study the problem of resource scheduling in *Grid computing Over Optical Burst Switching* (GoOBS) architecture. We consider two types of grid applications: *non-divisible load*—applications for which the workload can not be divided and must be processed sequentially; and *divisible load*—the applications for which the workload is arbitrarily or modularly divisible and can be processed parallelly [5]. Our objective is to minimize the resource blocking rate by selecting destinations *non-selfishly* to reduce the probability of *collision at destinations*. The selection of a destination by a request is called *non-selfish*, if the selected destination has sufficient resources available to simultaneously process one or more additional requests. Four heuristics based on the *non-selfish destination selection paradigm* are presented in this paper. Extensive simulations were performed to validate the effectiveness of the proposed heuristics. In our experiments, we find that among the proposed heuristics, the NFFD is most effective and reduces the resource blocking rate by 21% to 73% as compared to the best existing approach [2].

II. Related Work

OBS may be the most suitable architecture to support next-generation distributed applications that require low volume of data transfer [6]. In OBS, an ingress node assembles one or more data packets into a burst which is transmitted to the egress node. Each burst is accompanied by a control message known as burst header, which contains information for reserving network resources by the intermediate nodes. The burst header can be encapsulated in the burst [7] or transmitted separately over a dedicated control channel [8].

Grid computing Over OBS (GoOBS) architecture has been recently propagated in several research papers [1]. This has resulted in an effort by Open Grid Forum to standardize GoOBS. One of the main focuses of the researchers is to improve the performance of the gird applications by reducing the resource blocking rate and minimizing the data loss due to burst contentions in the OBS network. Different transmission techniques have been investigated to improve the performance of grid applications. It was reported that anycast [3] and many-cast [9] transmission techniques can significantly improve the performance of many grid applications.

Two main factors that can influence the performance of GoOBS are the selection of destination(s), and the selection of communication path(s) [2], [4]. Guerreiro *et al.* have investigated efficient path selection strategies for GoOBS [10]. The concept of dynamic load balancing to improve the performance of grid applications has been investigated in [11].

Kannasoot *et al.* proposed the "Sequential Task Anycast Scheduling" (STAS) problem to process the sequential dependent subtasks in GoOBS using anycast transmission [4]. She *et al.* proposed the "Multi-Resource Manycast" (MRM) problem to process independent requests in GoOBS using manycast transmission [2]. The experiments verified that the change in the state of the resources at a destination between the time of a request's selection of a destination and its arrival at the destination results in higher resource blocking rate [2], [4].

In this paper, we minimize the resource blocking rate in GoOBS by using the *non-selfish destination selection paradigm*. To the best of our knowledge, *non-selfish* scheduling strategies in GoOBS have not yet received much attention.

III. Problem Description

In this section, we describe the problem, its assumptions, and prove it is $\mathcal{NP}-$complete.

Preliminaries: We assume the network of grids is supported by an OBS architecture. A source selects the destination(s) to process a request. Each request is assembled into one or more bursts that are transmitted to the destination(s). The burst header is encapsulated in the burst [7] and is processed all-optically [12] by the intermediate nodes to route the burst.

Now, we define the concept of resource blocking for a request. There are two scenarios in which resources can be blocked for a request [2]. First, a burst containing a request may be dropped due to network contention in the OBS network. The resources requested by a dropped burst will be blocked for the request [2]. We call this kind of resource blocking *resources blocked due to network contention* (RBNC). Second, if a request can not be processed due to unavailability of sufficient resources at the destination candidates, then this kind of resource blocking is called *resources blocked due to resource contention* (RBRC). This includes the requests dropped at a source because no destination can be identified with sufficient available resources; and the requests blocked due to *collision at destinations*. The total resources blocked (TRB) is the sum of the total RBNC and the total RBRC [2].

The total resource blocking rate (TRB rate) is defined as the ratio of the TRB and the total resources requested (TRR). The total resource blocking rate due to network contention (RBNC rate) is defined as the ratio of the total RBNC and the TRR. Similarly, total resource blocking rate due to resource contention (RBRC rate) is defined as the ratio of the total RBRC and the TRR.

Network Model: A GoOBS network is represented by a simple, undirected, connected graph, $G = (V, E)$, where V represents the set of grid nodes and E represents the set of fiber links in the network. Node identifier, id is a unique number assigned to each grid node. A user request is denoted by $R(id, qty)$, where id is the source identifier, and qty is the amount of resources requested. The amount of resources available at each node is denoted by A_{id}.

Request Model: We assume a single type of resource [2], processing units (PUs) is available in the network. Each site has a non-zero number of initial resources. The number of available resources at a destination will change as reservations are made for an arrived request. Requests are processed on a first-come first-serve basis.

Each request with a *non-divisible* load is assembled into a burst. On the other hand, a request with *divisible* load is divided into a number of independent requests that are considered to be *non-divisible*. The number of independent requests is determined by the maximum number of resources a burst can request, which is bounded by the *Resources per Burst* (RpB) parameter. This kind of "Limit per Burst" [2] technique helps to minimize the resource blocking rate. Therefore, a request with *divisible* load $R(id, qty)$, $(qty > RpB)$ is divided into $\left\lceil \frac{qty}{RpB} \right\rceil$ number of *non-divisible* independent requests.

Problem Statement: Let $G = (V, E)$ be a GoOBS network. Given A_{id} for each node, and a set of user requests, $\mathscr{R}$. The objective is to minimize the resource blocking rate.

$\mathcal{NP}$ **Complete:** We prove that the resource scheduling problem in GoOBS is $\mathcal{NP}-$complete by reducing the bin packing problem to the resource scheduling problem in GoOBS. The bin packing decision problem (BPP) is defined as [13]:

Given: A finite set $U = \{u_1, u_2, \ldots u_n\}$ of items, a size $s(u) \in [0, 1]$ for each item $u \in U$, and a constant k.

Question: Is it possible to find a partition of U into disjoint subsets $U_1, U_2, \ldots U_k$, such that the sum of sizes of the item in each U_i is no more than 1.

The polynomial transformation of the BPP to the resource scheduling problem in GoOBS is: $k = |V|$; each item is a burst; $s(u)$ is the number of resources requested by a burst; the maximum number of resources available at a site is 1, which is the maximum size of a partition. We assume the network has an infinite bandwidth, and there is no network delay. Thus, each partition U_i corresponds to the set of bursts which can be processed at the site i in the GoOBS network.

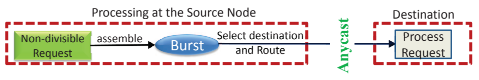
The BPP polynomially reduces to the resource scheduling problem in GoOBS. Thus, it is $\mathcal{NP}-$complete.

IV. *Non-Selfish Destination Selection Paradigm*

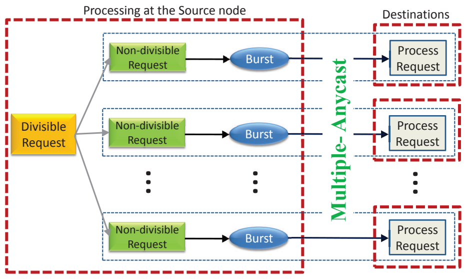
The objective of the *non-selfish destination selection paradigm* is to reduce the probability of *collision at destinations* and minimize the resource blocking rate or the TRB rate. In order to achieve our objective, we reduce both the RBNC and RBRC rates simultaneously.

The RBNC rate is minimized by reducing the network contention in the OBS network. The network contention can be minimized by reducing the length of the path traversed by a burst [2], [4]. Thus, a shortest path between the source and each of the destination nodes is used for transmitting respective bursts.

The RBRC rate is minimized by reducing the resource contention. To minimize the resource contention and overcome the *collision at destinations*, the selection of a destination by a request is done in such a way that the destination has sufficient resources available to simultaneously process one or more additional requests. We call this a *non-selfish* destination selection strategy which reduces the probability of *collision at destinations*. Thus, *non-selfish* destination selection strategies can help reduce both the resource and network contentions.



(a) Request with *non-divisible* load



(b) Request with *divisible* load

Fig. 1. *Non-Selfish Destination Selection Paradigm*

Requests with *non-divisible* loads: A request with *non-divisible* load (called a *non-divisible* request) is assembled into a burst at the source. The source selects a *non-selfish* destination, and sends the burst to the destination (see Fig. 1(a)). Each burst contains data to be processed, amount of resources (PUs) requested, holding (service) time of the resources, and the destination of the request. When the burst arrives at the destination, the requested resources will be reserved for the holding time of the request. If enough resources are not available, the request is blocked and no resources are reserved. If the burst is dropped due to network contention, no resources are reserved. If a source is unable to select a *non-selfish* destination, then the request is dropped at the source. The burst corresponding to the dropped request is also dropped at the source, and no resources are reserved.

Requests with *divisible* loads: A request with *divisible* load (called a *divisible* request) is processed at one or more destinations. Based on the value of RpB, a *divisible* request is divided into one or more independent requests considered to be *non-divisible*. The source assembles these independent requests into separate bursts and selects a *non-selfish* destination for each burst (see Fig. 1(b)). Thus, multiple-anycast transmission is used to transmit a *divisible* request. Our approach is different from the *hop and divide* approach presented in [2]. The destinations do not need to communicate with each other, as the requests are considered to be independent [2]. *Divisible* requests can not use manycast [9] or multicast schemes, as the bursts have different payloads.

The selection of a *non-selfish* destination by a source is based on the number of PUs needed by the respective requests to be scheduled, and the number of PUs available at all the sites. The information about the number of PUs available at a site is periodically broadcasted using a control packet [2], [4], [6]. The information about the number of PUs needed by the respective unscheduled requests at a source can be integrated into the control packet with negligible overhead.

A. *Non-Selfish Destination Selection Heuristics*

We present four destination selection heuristics by integrating the *non-selfish destination selection paradigm* with bin packing heuristics [14]. We only consider a priori division of arbitrarily or modularly divisible workloads. Our heuristics are different from existing approaches [2], [4]. The proposed heuristics based on the *non-selfish destination selection paradigm* (called *non-selfish* heuristics) are described below.

Best Fit Destination (BFD): A destination is called *best fit destination* for a request, if it has the *least* number of available resources, but sufficient to process not only this request but at least K more randomly selected requests from nodes with smaller id. In case of a tie, choose the nearest destination.

Worst Fit Destination (WFD): A destination is called *worst fit destination* for a request, if it has the *most* number of available resources, but sufficient to process not only this request but also K more randomly selected requests from nodes with smaller id. In case of a tie, choose the nearest destination.

Nearest First Fit Destination (NFFD): A destination is called *nearest first fit destination* for a request, if it is the nearest destination to the source, and has sufficient resources to process not only this request but also K more randomly selected requests from nodes with smaller id and within one hop from the destination. In case of a tie, choose a destination with the most number of available resources.

Random Select Destination (RSD): A randomly selected destination with uniform probability is called *randomly selected destination* for a request, if it has sufficient resources to process not only this request but also K more randomly selected requests from nodes with smaller id and within three hops from the destination. If a destination is not successfully selected in a fixed number of tries (number of nodes in the network), then the request is dropped at the source.

B. *Route Selection*

Floyd-Warshall algorithm is used to find the all pairs shortest paths [14]. A shortest path between a source and the destination is used to transmit the burst in the OBS network.

C. *An Illustrative Example*

We present an example to illustrate the proposed *non-selfish destination selection paradigm*. A GoOBS network is represented in Fig. 2(a). The node identifiers are represented inside the vertices (id A is smaller than id B). The number at each vertex represents the amount of resources (PUs) available at the site. The number at each edge represents the length of the link. Let us consider two *non-divisible* requests, $R(A, 3)$ and $R(B, 2)$ which are to be simultaneously scheduled.

If both requests selfishly select their closest destination [2], both of them will select site F as the candidate destination. This results in collision at the destination and only one of the requests can be successfully processed (Fig. 2(b)). However, if the requests *non-selfishly* select their destination using NFFD (K=1), then request $R(B, 2)$ selects site C as the candidate destination, since site C has sufficient available resources to

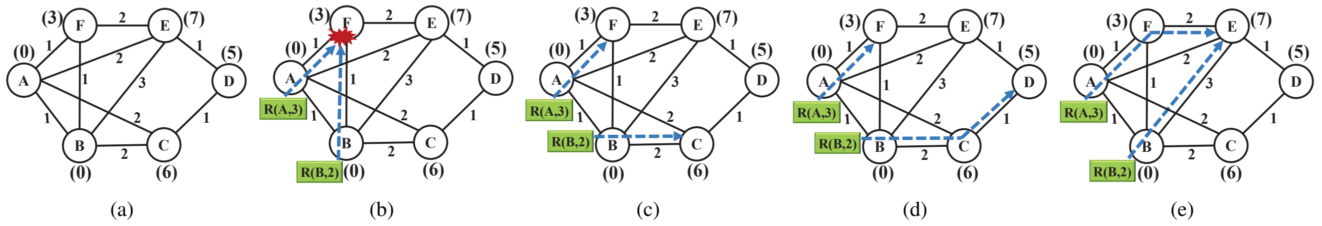


Fig. 2. (a) A GoOBS network; (b) Requests $R(A, 3)$ and $R(B, 2)$ selfishly selecting their closest destination (site F) lead to a collision; (c) *Non-selfish* selection of destinations for requests $R(A, 3)$ and $R(B, 2)$ (sites F and C respectively) using NFFD; (d) *Non-selfish* selection of destinations for requests $R(A, 3)$ and $R(B, 2)$ (sites F and D respectively) using BFD; (e) *Non-selfish* selection of destinations for requests $R(A, 3)$ and $R(B, 2)$ (site E) using WFD

simultaneously process both the requests (Fig. 2(c)). The *non-selfish* destinations for the requests using BFD (*K*=1) and WFD (*K*=1) are shown in Figs. 2(d) and 2(e) respectively.

V. Results and Discussion

In this section, we describe the experiments and analyze the results obtained. For comparative analysis of our heuristics, we use the best known existing approach for the problem framework considered in this paper. The following algorithm presented in [2], was reported to have the best performance.

Closest Destination First (CDF) [2]: A destination is called *closest destination first* for a request, if it is the nearest destination to the source, and has sufficient resources to process this request [2]. Ties are broken randomly.

A. Experimental Setup

We use the 16-node, 25-link NSFNET topology for simulation. There are two bidirectional fibers on each link and four wavelengths per fiber. Each fiber has a uniform transmission capacity of 10Gbps. There are no fiber delay lines, optical buffers or wavelength converters in the network. The *Last Available Unscheduled Channel with Void Filling* (LAUC-VF) [15] scheduling algorithm is used for bursts. A unique integer between 1 and 16 is selected with equal probability as the node identifier (id) for a site. Requests or tasks are generated with a Poisson distribution. The burst assembly time and the burst header processing time are 1ms and 20μs, respectively. The average length of the bursts and the average holding time of the resources are exponentially distributed with mean of 15MB and 5ms, respectively. The burst loss ratio is the ratio of the total number of bursts dropped due to network contention and the total number of bursts that attempted transmission. The default value of K is 1. The results represent an average of 20 runs and all results have 95% confidence level.

B. Requests with Non-divisible Loads

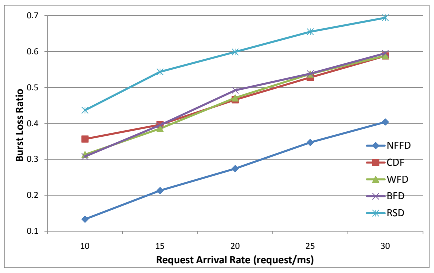
The average number of initial resources (PUs) available at each site and the average number of resources requested by a task is uniformly distributed between 10-12 units and 3-8 units, respectively. However, for the distribution of the sources of the requests among the nodes in the network, we experimented with both uniform and non-uniform distributions.

1) Uniform distribution of sources among the nodes in the network: The performances of different heuristics in terms of the burst loss ratio are compared in Fig. 3(a). NFFD has the least burst loss ratio among all the heuristics. Thus, NFFD is most effective in minimizing the network contention. Surprisingly, RSD has not so good performance in this experiment. This is due to its randomness in selection of a *non-selfish* destination. A randomly selected destination for a request may be far away from the source resulting in an increased network contention. The number of bursts in the network increases with an increase in the request arrival rate. This results in an increase in the network contention. Thus, the burst loss ratio increases with an increase in the request arrival rate.

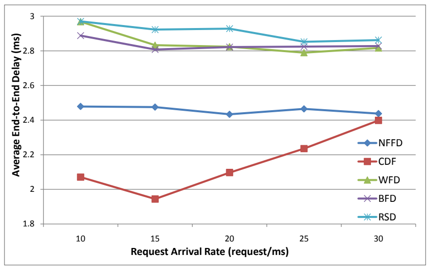
In Fig. 3(b), the performances of the heuristics are compared in terms of the average end-to-end delay. A CDF destination [2] for the request is the nearest to the source. Thus, CDF [2] has the least average end-to-end delay. As expected, a nearest first fit destination for the request is nearer than those selected using WFD, BFD, and RSD heuristics for the request. Interestingly, a best fit destination and a worst fit destination for the request are nearer than a randomly selected destination. This is reflected in the highest average end-to-end delay values of RSD. So, RSD has higher burst loss ratio in Fig. 3(a).

The RBNC rates of the heuristics are compared in Fig. 3(c). NFFD has the least RBNC rate as compared with other heuristics. As observed in Fig. 3(a), NFFD is most efficient in reducing the network contention. The *non-selfish* heuristics can identify requests likely to be blocked due to *collision at destinations*. These requests and their corresponding bursts are both dropped at the source reducing the network contention. So, even though a CDF destination [2] is closer than a nearest fit first destination for a request, CDF [2] still has higher burst loss ratio and higher RBNC rate as compared to NFFD. Similarly, RSD has comparable RBNC rate with CDF [2], even though it has higher burst loss ratio than CDF [2].

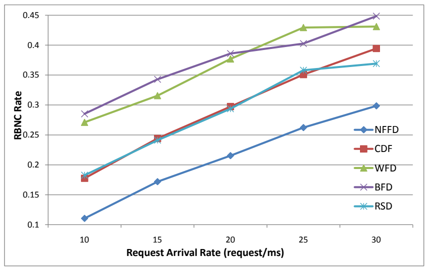
The RBRC rates of NFFD, WFD, and BFD heuristics have comparable values in Fig. 3(d). The *non-selfish* heuristics have lower RBRC rates as compared with CDF [2], except for RSD. Randomly selecting a *non-selfish* destination is probably no better than selfishly selecting the closest destination with sufficient resources available. RSD has high request dropping at the source which reduces the RBNC rate but adversely affects the RBRC rate. To further investigate this behavior of RSD, we modified its definition to randomly select a destination for a request such that the destination has sufficient resources to process not only this request but also K more randomly selected requests from nodes with smaller id and within *two* hops from the destination. This improves the RBRC rate but adversely affects the RBNC rate. Thus, RSD can not simultaneously improve both the RBNC and RBRC rates as



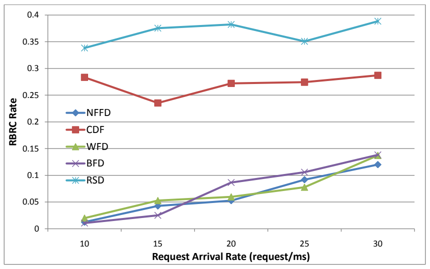
(a) Burst Loss Ratio



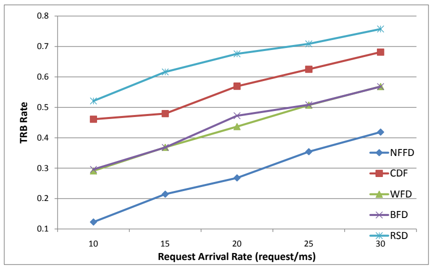
(b) Average End-to-End Delay



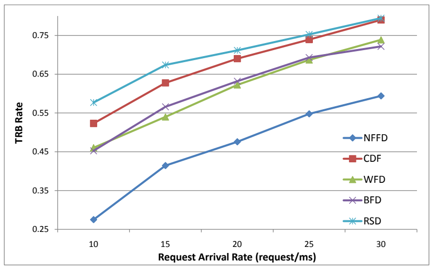
(c) RBNC Rate



(d) RBRC Rate



(e) TRB Rate



(f) TRB Rate for Normal distribution of sources

Fig. 3. Requests with *Non-divisible* Loads

improvement in one degrades the performance of the other.

In Fig. 3(e), NFFD has the lowest TRB rate. The *non-selfish destination selection paradigm* indeed minimizes the resource blocking rate, as NFFD, WFD, and BFD have lower TRB rates as compared to CDF [2]. As explained earlier, random selection of a *non-selfish* destination does not improve the RBRC rate due to the high request dropping at the source. This results in not so good performance of RSD in reducing the TRB rate. The RBNC rates of WFD and BFD are higher than the RBNC rate of RSD in Fig. 3(c). But, the TRB rates of WFD and BFD are lower than the TRB rate of RSD. So, WFD and BFD have lower request dropping at the source as compared to RSD, which results in lower RBRC rates of WFD and BFD as compared to the RBRC rate of RSD.

2) Non-uniform distribution of sources: We study the impact of a Normal distribution of sources (*mean*=*8*, *variance*=*2*; where $|V| = 16$) on the TRB rate in Fig. 3(f). Experiments with other values of *mean* and *variance* have similar results. A non-uniform distribution of sources increases the probability of *collision at destinations* as several requests tend to choose the same destination. This increases the TRB rate. The performances of NFFD, WFD, and BFD are better than CDF [2] in this experiment. Interestingly, the TRB rate of RSD is comparable with CDF [2] under higher request arrival rate.

C. Requests with Divisible Loads

The source nodes of the tasks or requests are uniformly distributed among all the nodes in the network. Experiments with non-uniform distribution of sources have similar trends as presented in Section V-B2. The average number of initial resources (PUs) available at each site and the average number of resources requested by a task is uniformly distributed between 10-15 and 20-30 units, respectively. The arrival rate of requests with *divisible* load is assumed to be 30 requests/ms.

In Fig. 4(a), we compare the performances of different heuristics in terms of the burst loss ratio. The number of bursts generated for a *divisible* request is inversely proportional to the value of RpB, i.e. for a *divisible* request, an increase in the value of RpB decreases the number of bursts. Thus, the burst loss ratio decreases with an increase in the value of RpB.

The average end-to-end delay of the heuristics are compared in Fig. 4(b). We see a trend similar to that in Fig. 3(b), i.e. a CDF destination [2] for the request is nearest to the source. Thus, CDF [2] has the lowest average end-to-end delay.

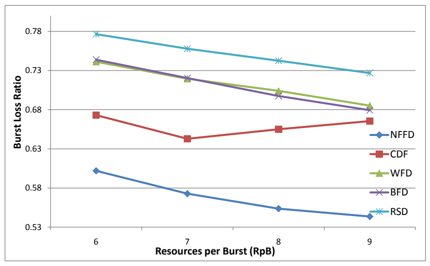
We observe in Fig. 4(c) that the RBNC rates of WFD, BFD, and RSD are higher than the RBNC rate of CDF [2] due to the selection of relatively farther destinations. Even though the burst loss ratio of RSD is higher than that of WFD and BFD, the RBNC rate of RSD is slightly lower than the RBNC rates of WFD and BFD. This is due to a higher request dropping at the source which reduces the network contention. Similarly, due to a high request dropping at the source when the value of RpB is 9, the RBNC rates of WFD, BFD, and RSD are slightly lower than the RBNC rate of CDF [2].

The resources in the network are more likely to be fragmented or scattered at higher values of RpB. This results in higher RBRC rates of the heuristics at higher values of RpB in Fig. 4(d). Thus, an increase in fragmentation of the resources also increases the request dropping at the source.

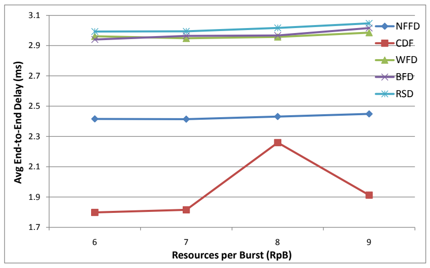
The results in Fig. 4(e) show an interesting trade-off between the value of RpB and the TRB rate. An increase in the value of RpB results in higher fragmentation of the resources. This causes higher request dropping at the source but lowers the network contention. On the other hand, a decrease in the value of RpB increases the number of bursts. This increases the network contention but lowers the fragmentation of the resources. Therefore, we need to simultaneously optimize the number of bursts generated from a *divisible* request while minimizing the value of RpB. In our experiment when the value of RpB is 7, all heuristics have their lowest TRB rate.

D. Discussion

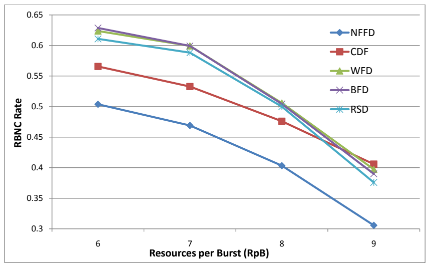
An analysis of the experimental data shows that the NFFD, BFD, and WFD have lower resource blocking rates than CDF



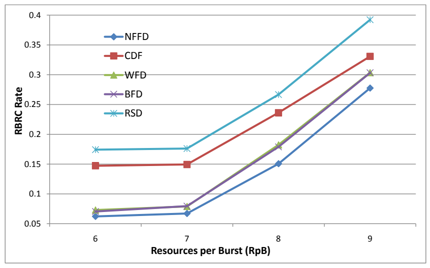
(a) Burst Loss Ratio



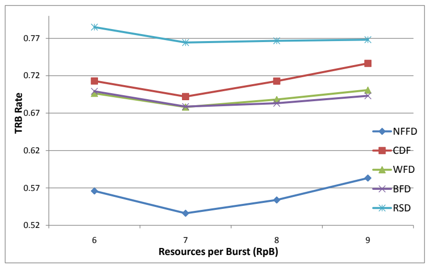
(b) Average End-to-End Delay



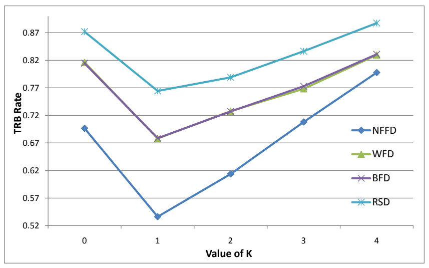
(c) RBNC Rate



(d) RBRC Rate



(e) TRB Rate



(f) TRB Rate with varying K (RpB=7)

Fig. 4. Requests with *Divisible* Loads

[2] (see Figs. 3(e), 4(e)). The NFFD is most effective in minimizing the TRB rate. Analyzing the results shown in Fig. 4(e) with RpB=6 and in Fig. 3(e) with 10 requests/ms, we observe that in our experiments the NFFD reduces the TRB rate by 21% to 73% respectively as compared to CDF [2]. The NFFD reduces the probability of *collision at destinations* and minimizes the RBRC rate (see Figs. 3(d), 4(d)). It also minimizes the network contention and has lowest burst loss ratio (see Figs. 3(a), 3(c), 4(a), 4(c)). Overall, among the heuristics based on the *non-selfish destination selection paradigm*, the NFFD has the best performance.

1) Changing the value of K: The impact of varying the value of K on the performance of the *non-selfish* heuristics is studied in Fig. 4(f). We observe that the performance of the *non-selfish* heuristics is adversely affected by an increase in the value of K. This is because increasing the value of K implies an over optimistic view of the destination selection approach. The TRB rate of NFFD is most significantly affected by an increase in value of K. On the other hand when K=0, our heuristics degenerate into a *selfish* strategy and thus results in higher TRB rate. We observe similar trends for *non-divisible* requests. Thus, the *non-selfish* heuristics have best performance when the value of K is 1.

2) Changing the experimental setup: We modified the experimental setup by changing the distributions, the values of the parameters, and the burst scheduling strategy. For example, normal and exponential distributions instead of uniform; and Just-Enough-Time (JET) [8] protocol in place of LAUC-VF. The NFFD heuristic has the best performance in these experimental setups with *non-divisible* and *divisible* requests. Details of these experiments are omitted due to space limitation.

VI. Conclusion

We propose a *non-selfish destination selection paradigm* to minimize the resource blocking rate in GoOBS. Grid applications with *non-divisible* and *divisible* loads are considered in this paper. Anycast and multiple-anycast transmissions are used to transmit requests with *non-divisible* and *divisible* loads respectively. Four heuristics based on the *non-selfish destination selection paradigm* are presented to minimize the resource blocking rate. Extensive simulations were performed to validate the effectiveness of these heuristics. Among the proposed heuristics, the NFFD heuristic is most effective in minimizing the resource blocking rate. Compared to the best existing approach, the NFFD reduces the resource blocking rate by 21% to 73% in our experiments.

References

[1] D. Simeonidou, *et al.*, "Dynamic optical-network architectures and technologies for existing and emerging grid services," *J. Lightw. Technol.*, vol. 23, no. 10, pp. 3347-3357, 2005.
[2] Q. She, X. Huang, N. Kannasoot, Z. Qiong, J. Jue, "Multi-Resource Manycast over Optical Burst Switched Networks," *IEEE ICCCN*, pp. 222-227, 2007.
[3] K. Lu, T. Zhang, A. Jafari, "An Anycast Routing Scheme for Supporting Emerging Grid Computing Applications in OBS Networks," *IEEE ICC*, pp. 2307-2312, 2007.
[4] N. Kannasoot, A. Patel, J. Jue, "Sequential Task Anycast Scheduling in Optical Burst Switched Networks," *IEEE ICC*, pp. 1-5, 2010.
[5] X. Lin, *et al.*, "Real-Time Scheduling of Divisible Loads in Cluster Computing Environments," *JPDC,* vol. 70, no. 3, pp. 296-308, 2010.
[6] W. Adlan, T. El-Gorashi, J. Elmirghani, "Anycast routing in OBS based Grid networks under heterogeneous traffic," *IEEE ICTON*, pp. 1-4, 2009.
[7] M. De Leenheer, *et al.*, "An OBS-based grid architecture", *IEEE GLOBECOM Workshops*, pp. 390-394, 2004.
[8] J. Jue, V. Vokkarane, "Optical Burst-Switched Networks", *Springer, Optical Networks Series*, 2005.
[9] B. Bathula, R. Bikram, V. Vokkarane, S. Talabattula, "Quality-of-Transmission-Aware Manycasting Over Optical Burst-Switched Networks," *IEEE/OSA JOCN*, vol. 2, no. 10, pp. 820-830, October 2010.
[10] M. Guerreiro, *et al.*, "Path selection strategy for consumer grid over OBS networks," *IEEE ICTON*, vol. 3, pp. 138-141, 2008.
[11] R. Bikram, V. Vokkarane, "Dynamic load-balanced manycasting over optical burst-switched (OBS) networks," *OFC*, pp. 1-3, 2009.
[12] Y. Liu, *et al.*, "Ultrafast all-optical signal processing: towards optical packet switching," *Proc. SPIE*, vol. 6353, pp. 635312-8, 2006.
[13] M. R. Garey and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of $\mathcal{NP}-$Completeness," *W. H. Freeman*, 1979.
[14] T. Cormen, *et al.*, "Introduction to Algorithms," *MIT Press,* 3^{rd} *ed.*
[15] Y. Xiong, M. Vandenhoute, H. Cankaya, "Control architecture in optical burst-switched WDM networks," *Selected Areas in Communications, IEEE Journal on*, vol. 18, no. 10, pp. 1838-1851, Oct 2000.