FeatherWeight: Low-cost Optical Arbitration with QoS Support

Yan Pan*
Globalfoundries Inc.
Malta, NY
panyan@gmail.com

John Kim
Web Science Technology Division & Dept. of Computer Science
KAIST
Daejeon, Korea
jjk12@kaist.edu

Gokhan Memik
Dept. of Electrical Eng. and Computer Science
Northwestern University
Evanston, IL
memik@eecs.northwestern.edu

ABSTRACT

The nanophotonic signaling technology enables efficient global communication and low-diameter networks such as crossbars that are often optically arbitrated. However, existing optical arbitration schemes incur costly overheads (e.g., waveguides, laser power, etc.) to avoid starvation caused by their inherent fixed priority, which limits their applicability in power-bounded future many-core processors. On the other hand, quality-of-service (QoS) support in the on-chip network is becoming necessary due to an increase in the number of components in the network. Most prior work on QoS in on-chip networks has focused on conventional multi-hop electrical networks, where the efficiency of QoS is hindered by the limited capabilities of electrical global communication. In this work, we exploit the benefits of nanophotonics to build a lightweight optical arbitration scheme, FeatherWeight, with QoS support. Leveraging the efficient global communication, we devise a feedback-controlled, adaptive source throttling scheme to asymptotically approach weighted max-min fairness among all the nodes on the chip. By re-using existing datapath components to exchange minimal global information, FeatherWeight provides freedom from starvation while resulting in negligible ($< 1\%$) throughput loss compared to the best-effort baseline optical arbitration. In addition, FeatherWeight provides strong fairness, performance isolation, and differentiated service for a wide range of traffic patterns. Compared to state-of-art optical arbitration schemes, FeatherWeight reduces power consumption by up to 87% while reducing execution time by 7.5%, on average, across SPLASH-2 and MineBench traces, and improving throughput on synthetic traffic patterns by up to 17%.

*This work was carried out when Yan Pan was associated with Northwestern University.

MICRO'11 December 3-7, 2011, Porto Alegre, Brazil

Categories and Subject Descriptors

C.1.2 [**Computer Systems Organization**]: Multiprocessors—*Interconnection architectures*; B.4.3 [**Hardware**]: Interconnections—*Topology*

General Terms

Design, Performance

Keywords

Interconnection Networks, Nanophotonics, Arbitration

1. INTRODUCTION

As technology advances, a variety of components are integrated onto a single chip: together with the increasing number of cores, an increasing variety of components are being integrated. For example, the Intel Nehalem architecture integrates memory controllers into the processor package [16], while the AMD Fusion family of APUs [1] combine GPUs and CPUs together. Thus, efficient and fair communication among all the different components will be critical to the overall performance of the processor. In addition, QoS support is becoming necessary for on-chip networks: in a network with several node types, each may require a different service level.

Recent progress in silicon nanophotonics [2, 34, 35] has provided an attractive communication fabric for future many-core processors. Both Intel [9] and IBM [8] have announced prototype chips that demonstrate the huge potential in this emerging technology. With its high bandwidth density, low latency and repeater-less communication, this emerging technology is especially efficient for long-range on-chip communication [13, 27, 31] and global arbitration [26, 30]. However, one common problem in existing optical arbitration is the fixed priority of each node; particularly, extra effort has to be placed to avoid starvation. Previous works have used longer waveguides [26] and broadcast buses [30], which limit the scalability of the overall architecture and cause significant power overhead. On the other hand, if nanophotonic signaling becomes the backbone of future on-chip networks, QoS support in nanophotonic networks will be necessary to provide both performance isolation and differentiated service [17]. However, existing optical arbitration schemes only focus on fairness and lack the flexibility to support differentiated service. Thus, a novel optical arbitration scheme with

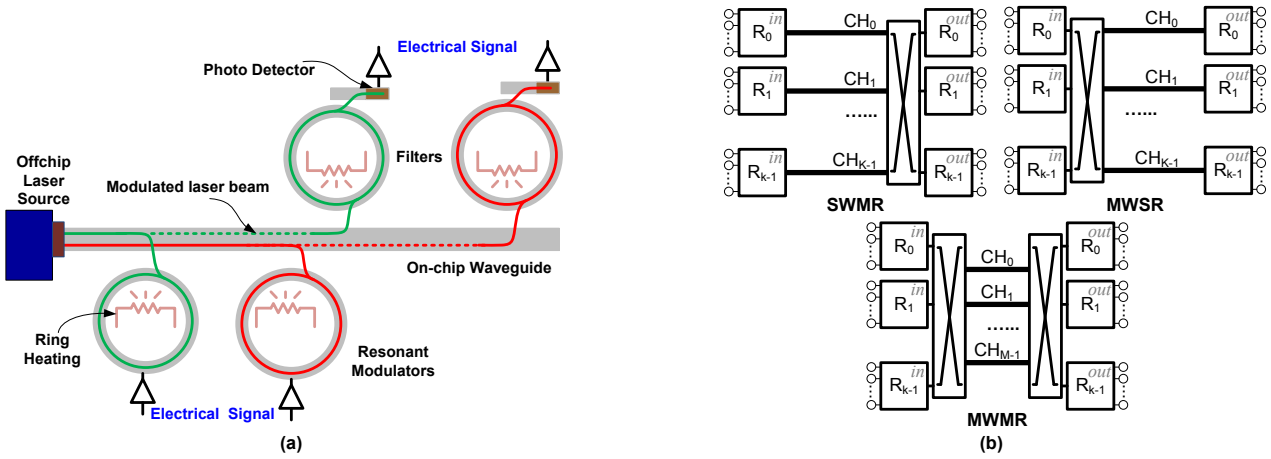

Figure 1: (a) Nanophotonic components and (b) logical architecture of optical crossbars [26].

QoS support is highly needed for future nanophotonic on-chip networks.

In this paper, we propose FeatherWeight, a lightweight optical arbitration scheme with QoS support. FeatherWeight leverages the efficient global communication capabilities of nanophotonics to build a feedback control system to adaptively throttle the network nodes and asymptotically approach a weighted max-min [11] fairness among all the nodes. Starvation is avoided as QoS is enforced. The weight of each node can be assigned by the run-time system to provide differentiated service levels. FeatherWeight exchanges minimal amount of information using the data channels, avoiding any extra optical hardware. To adapt to the dynamics of on-chip network traffic, dynamic adjustments to the bandwidth allocation is carried out once every epoch, so as to avoid excessive computational complexity. Compared to prior work [26, 30], significant amount of hardware and power reduction is achieved, which greatly improves the scalability and applicability of optical arbitration.

In summary, the contributions of this paper include:

- We propose FeatherWeight – a lightweight, optical arbitration scheme based on feedback-controlled, adaptive source throttling with support for QoS.
- We leverage the low latency and efficient global communication of nanophotonics and existing waveguides to significantly reduce the cost (both area and power) over prior approaches to global optical arbitration.
- Exploiting the lightweight, optical arbitration, we provide QoS support in the arbitration with minimal additional overhead.
- We evaluate FeatherWeight and compare it against state-of-art alternatives. We demonstrate that FeatherWeight achieves high performance optical arbitration while providing robust QoS support at low cost.

The remainder of this paper is organized as follows. In Section 2, we present background information and related work. The details of the proposed FeatherWeight scheme is presented in Section 3. We evaluate the performance and cost of FeatherWeight in Section 4. Section 5 summarizes and concludes the paper.

2. BACKGROUND AND RELATED WORK

2.1 Nanophotonics and Optical Crossbars

Recent progress in silicon nanophotonics [2, 34, 35] promises a new on-chip communication fabric with low latency and high bandwidth density. The basic components in nanophotonic signaling are illustrated in Figure 1(a). Laser beams of various wavelengths are generated off-chip and coupled onto an on-chip waveguide, which traverses different nodes on the network. Wavelength-selective resonant modulators [34] are built to modulate electrical signals onto specific wavelengths, which can then be filtered, detected, and converted back into the corresponding electrical signal [35]. The key component is the ring resonator that enables multiple logical data channels to be implemented on different wavelengths in the same physical waveguide. The power consumption of such an optical channel includes the electrical power to modulate and demodulate the signals, the laser power, and the ring heating power [2] to fine-tune the resonant wavelengths.

Different topologies have been proposed to utilize such nanophotonic signaling [10, 14, 15, 27, 28, 31]. Since nanophotonics provides low latency and does not require repeater insertion, it is ideal for implementing global crossbars [26, 31]. Compared to alternative topologies (e.g., torus [28], etc.), such crossbar topology is more scalable and cost-effective as the network diameter is reduced [12]. Vantrease et al. [31] proposed the Corona architecture with a radix-64 crossbar, while hierarchical topologies like Firefly [27] leverages nanophotonic crossbars and can further increase the network size.

Pan et al. [26] categorized the 3 alternatives to organize the optical channels to build nanophotonic crossbars, namely SWMR, MWSR, and MWMR, as shown in Figure 1(b). SWMR, or Single-Write-Multiple-Read crossbars uses the optical channel for each sender to transmit its packets, while the arbitration is localized to the receiver side and done electrically. MWSR, or Multiple-Write-Single-Read crossbars, on the other hand, dedicates a channel for each router to receive packets, and global arbitration is needed to resolve conflicts among the senders. MWMR, or FlexiShare optical crossbar, combines both arbitration and allows fewer channels to be used for better power efficiency under unbalanced traffic. To enable these crossbars, especially the MWSR and MWMR crossbars, efficient global arbitration is critical.

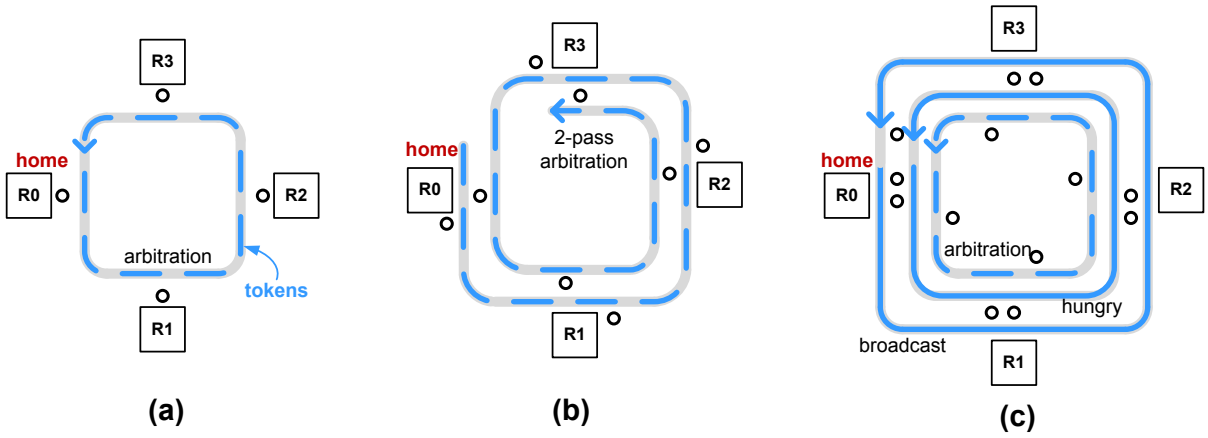

Figure 2: Hardware for (a) baseline optical arbitration, (b) 2-pass Token Stream, and (c) Fair Slot. Waveguides carrying laser (e.g., optical tokens) traverse all the routers for information delivery.

2.2 Slot-based Optical Arbitration

Two time slot-based optical arbitration schemes have been recently proposed for arbitrating global optical crossbars. Vantrease et al. [30] proposed a *Fair Slot* arbitration scheme for arbitrating MWSR nanophotonic crossbars; while Pan et al. [26] introduced *2-pass Token Stream* that can be used for MWSR and MWMR crossbars. Both schemes arbitrate on a time-slot basis, and are more efficient than the previously proposed *Token Ring* arbitration scheme [31].

2.2.1 *Baseline Optical Arbitration (Fixed Priority, Starvation-prone)*

Both Fair Slot and 2-pass Token Stream have a common *baseline optical arbitration* scheme [1] as illustrated in Figure 2(a). Arbitration is carried out on a time-slot basis, where units of optical energy (i.e., optical tokens) are used to represent the privilege to occupy units of resources (e.g., a time slot on the data channel). A waveguide that carries the optical tokens traverses all the nodes in a specific order, and each node has a ring resonator that can be turned on to couple the optical tokens off the waveguide. Since at most one node can grab a specific token an arbitration is achieved among all the nodes.

However, as the tokens are injected at a specific point (i.e., the *home* node) and take a fixed route (decided by the physical layout of the waveguide) across all the nodes, nodes closer to *home* have higher priority to grab a token. A busy node close to *home* may consume all the tokens, starving the farther downstream nodes.

2.2.2 *2-pass Token Stream*

To solve the potential starvation problem, Pan et al. [26] extend the baseline optical arbitration with a starvation avoidance scheme. Instead of having the 1-pass token waveguides, the waveguides are routed to pass each node twice, as shown in Figure 2(b). In the first pass, each token is dedicated to a different node, guaranteeing a minimum amount of bandwidth for each node. If a token is not grabbed by its dedicated node in the first pass, any node can grab it in the second pass. This scheme keeps the high bandwidth utilization of the baseline scheme and avoids starvation, but does not guarantee strict fairness: the nodes closer to *home* still stand higher chance to grab second-pass tokens. In addition, the 2-pass token traversal increases laser power consumption and limits the scalability of the arbitration scheme.

[1]This arbitration scheme is referred to as *Time Slot* and *1-pass Token Stream* by Vantrease et al. [30] and Pan et al. [26], respectively. For simplicity, in this work, we refer to it as the "baseline optical arbitration".

2.2.3 *Fair Slot*

Vantrease et al. [30], on the other hand, employs a starvation detection and recovery scheme in their *Fair Slot* optical arbitration, as illustrated in Figure 2(c). Besides the 1-bit channel for tokens used in the baseline optical arbitration, a separate channel is used to detect if any node has not had access to the data channel for some time (i.e., "hungry" or starving). This hungry signal is efficiently implemented as an optical logic OR – any node can couple the laser energy off this channel to indicate that it is hungry. In addition, an optical broadcast channel is used by the *home* node (i.e., token injection point) to inform all the nodes the change of operation state.

Initially, all the nodes start in "plenty" mode, when they can grab tokens freely until at least one node becomes hungry. The hungry node then informs the home node using the hungry signal. The home node, upon detecting the hungry signal, broadcasts a signal to all nodes indicating the switch to "famine" mode (i.e., starvation recovery mode), where only the hungry nodes can take turns to flush existing packets in their input queues. Hungry nodes who finish flushing packets will stop coupling the hungry signal and be suspended to give downstream nodes the chance to flush their packets. When all the hungry nodes have sent their packets, the home node broadcasts another signal indicating the switch back to plenty mode.

This scheme solves the starvation problem and approximates max-min fairness under heavy traffic. However, significantly more hardware is added, as can be seen in Figure 2. These extra hardware, especially the broadcast bus, results in high power consumption and limits the scalability of the system. In addition, at the end of each famine mode, all the nodes are suspended, and the tokens in-flight will be wasted, causing lower throughput of the system.

In this work, we propose FeatherWeight, a lightweight optical arbitration scheme with freedom from starvation. In

Scheme	Free of Starvation	Hardware Cost	Power Cost	Fairness	Differentiated Service
baseline [30, 26]	No	Low	Low	No	No
2-pass [26]	Yes	High	High	No	No
Fair Slot [30]	Yes	High	High	Yes	No
FeatherWeight	Yes	Low	Low	Yes	Yes

Table 1: Comparison of optical arbitration schemes

addition, strong QoS support (i.e., fairness and differentiated service) is provided even with the reduced hardware cost. A comparison of FeatherWeight against existing optical arbitration schemes is summarized in Table 1.

2.3 Quality of Service

Quality of Service (QoS) is the capability to provide resource assurance and service differentiation in a network [32]. QoS is essential when the shared resources are limited, and hence a certain form of regulation has to be installed to provide *performance isolation* and *differentiated service* [17]. *Performance isolation* requires that the resources (e.g., bandwidth) achieved by each node is isolated from the concurrent activities in the system while *Differentiated service* provides the flexibility to allocate the resources in varied proportions.

As the complexity of processors grows, a variety of resources are integrated (e.g., memory controller) and shared among competing components and processes. This calls for proper QoS support in resource arbitration. Several works studied QoS support for centralized processor resources like memory controllers [22] and cache banks [23]. Resource level QoS support cannot be guaranteed if QoS is not enabled in the on-chip network. Thus, recently researchers have focused on QoS support in conventional, multi-hop electrical networks [7, 17, 24]. QoS schemes proposed for large IP networks (e.g., Weighted Fair Queuing (WFQ) [5] and Virtual Clock [36]) achieve similar goals but have to be adapted to on-chip network constraints. For example, WFQ assumes that each flow is leaky bucket constrained, which is not applicable for adversarial on-chip network traffic pattern. Moreover, complexity and cost concerns also prompt customized on-chip QoS schemes, like Globally-Synchronized Frames (GSF) [17] and Preemptive Virtual Clock (PVC) [7], to be proposed. But as electrical signaling is inefficient for global communication, overheads in terms of lost throughput and/or wasted network resources are incurred. Furthermore, all these schemes address local arbitration and are not applicable for QoS in *global* arbitration.

2.4 Leveraging Nanophotonics for QoS

QoS support can also benefit from the nanophotonics technology. First, nanophotonics allows us to quickly exchange global information. Having the information of remote nodes easily accessible eliminates the need to make assumptions, which is commonly done in existing QoS schemes [17, 7], making the scheme more adaptive. Second, the network diameter can be significantly reduced. As compared to a multi-hop network where the QoS guarantee often has to be maintained by carefully adjusting multiple arbitration (e.g., PVC [7]), QoS support in a single arbitration is sufficient to meet the requirements of QoS in a crossbar topology. This potentially simplifies the problem, though the arbitration itself is distributed and requires special designs. In addition, QoS schemes often require storage of bandwidth consumption information. With the 1-hop network implemented in nanophotonics, this overhead is minimized as intermediate routers are removed. Recently, Ouyang et al. [25] proposed a frame based QoS scheme to enhance the Fair Slot optical arbitration scheme [30] with differentiated service, but added even more optical hardware (unicast completion ring & broadcast frame switching ring). In the proposed FeatherWeight optical arbitration scheme, we aim to provide the QoS guarantees while *significantly reduce* the optical hardware and power cost.

3. FEATHERWEIGHT OPTICAL ARBITRATION SCHEME

In FeatherWeight, weighted max-min fairness [11] is achieved with the hardware cost close to a baseline optical arbitration, as shown in Figure 3(a). A single QoS controller is used for each data channel (or each node). We exploit the waveguide layout and place the controllers at the end of the waveguides at the *home* node. Thus, no additional physical connectivity is required.

Max-min fairness is a widely accepted fairness definition in networking, where all the requesters with low demand will be satisfied, and all the high-demand nodes will get the same share of the resource. To provide differentiated service, we implement the generalized form of max-min fairness, with each node given a weight (W_i). Max-min fairness is then satisfied among the nodes in terms of their normalized resource consumption (i.e., resource consumption normalized to their weights). FeatherWeight adaptively adjusts itself according to the changing traffic patterns through a feedback control mechanism. High resource utilization can thus be maintained. For clarity, we summarize all the parameters used in this section in Table 2.

3.1 Overview: Adaptive Source Throttling

To solve the starvation problem in the baseline optical arbitration scheme, a quota (i.e., the maximum number of tokens each node can grab) is assigned to each node over a period of time (epoch). Any node that has consumed its dedicated quota in the epoch will be throttled and prevented from grabbing more tokens. If fixed quota is used, fairness and differentiated service can be achieved, but this results in poor resource utilization as the unused quota of low-demand nodes cannot be used by other high demand nodes.

In FeatherWeight we improve the resource utilization by a feedback control scheme, which dynamically changes the quota according to the request patterns. Minimal per-node token consumption information is shared globally among the nodes to make proper quota adjustment. Leveraging the efficient global communication and low latency of nanophotonics, the feedback system can efficiently track the change of the traffic patterns and achieve low hardware cost and high resource utilization while meeting the QoS objectives.

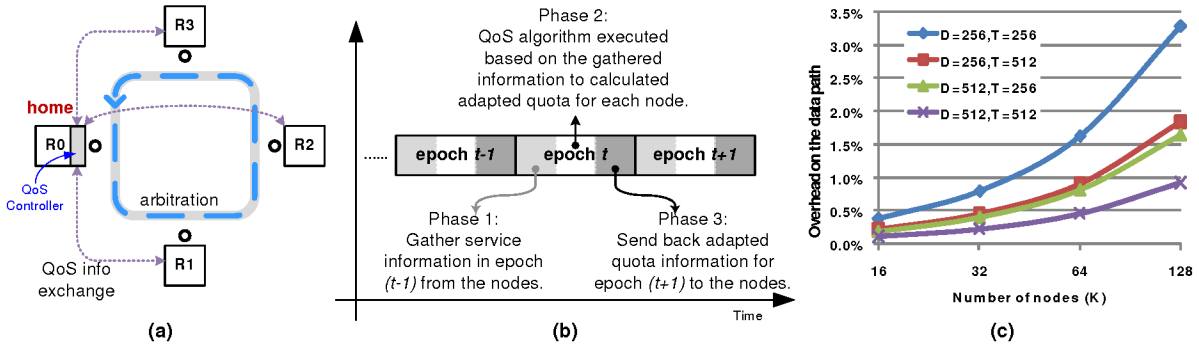

Figure 3: FeatherWeight scheme. (a) optical arbitration hardware for the data channels of R0, (b) timing diagram for an epoch, (c) bandwidth overhead for D-bit datapath and epoch size of T.

K	Total number of nodes in the arbitration (i.e., arbitration radix).
T	Size (number of cycles) of an epoch, over which quota adjustments are made.
W_i	Service weight of node i to indicate differentiated service levels.
A_i^t	Achieved service (i.e., number of tokens) by node i in the t-th epoch.
b_i^t	Busy status of node i in the t-th epoch.
Q_i^t	Adaptive service quota for node i in the t-th epoch.
C_i^t	Normalized accumulated achieved service by node i up to the t-th epoch.
$\bar{C}^t$	Average C_i^t among busy nodes of the t-th epoch.
h_i^t	High demand status of node i in the t-th epoch.
B_i^t	Base quota calculated based on weighted max-min fairness.
X_i^t	Adjustment quota to reflect the throttling of over-achieving nodes.
S^t	Total amount of resource to be shared among the high-demand nodes in epoch t.
α	A coefficient to control the service level of the high-demand nodes.
β	A coefficient to control the feedback strength of node throttling.
F	Time frame to reset the service statistics counters C_i^t to avoid excessive history effect.

Table 2: Summary of QoS Parameters

3.2 Enhanced Optical Arbitration in FeatherWeight

Consider a FeatherWeight system with 4 nodes, as shown in Figure 3(a). To support fairness and differentiated service, we use the same optical hardware as the *baseline optical arbitration*, but exchange minimal QoS information among the routers, as indicated by the dotted lines. There are three additions to the baseline optical arbitration: (a) bookkeeping at each node to gather service (bandwidth consumption) information; (b) QoS information exchange; and (c) a QoS controller, which executes the algorithm to calculate the quota based on global service information.

3.2.1 *Book-keeping at each node*

We divide time into *epochs* of T cycles each. Each node i monitors its *achieved service* (A_i^t), i.e., the number of claimed tokens, within each epoch t. For source throttling, a *service quota* (Q_i^t) is set for each node i in the t-th epoch (i.e., the maximum number of tokens it can grab in the epoch). With an epoch size of T, A_i^t and Q_i^t require $(\log_2 T)$-bit storage at each node.

Another important piece of information is whether a node is satisfied with its achieved amount of service within an epoch. We define a node i to be *busy* ($b_i^t = 1$) for an epoch t if it has at least one packet pending to be sent for every cycle in the epoch. Note that this definition is more relaxed than the "hungry" state defined in the Fair Slot scheme [30].

3.2.2 *QoS information exchange*

To achieve the QoS adjustments in a distributed optical arbitration, there has to be global communication that aligns the service levels (i.e., token consumption) at different nodes. In FeatherWeight, this is done in an organized manner, as shown in Figure 3(b). Each epoch t consists of 3 phases: in **Phase 1**, each node i sends its A_i^{t-1} and b_i^{t-1} values for the past epoch to a QoS controller[2], which is conveniently located at the end of the data channels. The controller then uses the collected global information to calculate the updated quota for each node in **Phase 2**. After the new quotas have been calculated, they will be propagated back to all the nodes in **Phase 3** and will be applied in epoch $t+1$. Note the adoption of nanophotonics greatly simplifies this information collection/distribution process, as no complicated wiring is needed.

This QoS information exchange is lightweight. For an epoch size of T cycles, A_i^{t-1} and Q_i^{t+1} are both $\log_2 T$-bit wide and b_i^t is a single bit. Hence, for an optical arbitration among K nodes, $K \times (\log_2 T + 1)$ bits have to be propagated from the nodes to the controller in each epoch, and $K \times \log_2 T$ bits have to be sent in the other direction. The total bandwidth needed is $(2K \log_2 T + K)/T$. If we use the optical data channel (e.g., re-using slots on the data

[2] The QoS controller can be implemented as a separate microcontroller or it can be embedded in the *home* node for each token channel (e.g., embedded in $R0$ in Figure 3(a)).

channel that is to be optically arbitrated in an MWSR optical crossbar) available in a nanophotonic crossbar to send such information, with a datapath width of $D = 512$ and $T = 256, K = 64$, the bandwidth overhead for the data bus is $(2K \log_2 T + K)/(DT) = 0.83\%$. A full analysis of bandwidth overhead under different scenarios is shown in Figure 3(c). If necessary, this overhead can be further reduced by utilizing coarser grain quota and achieved service levels[3]. Overall, the QoS information exchange poses negligible overhead for an optical datapath with, typically, abundant bandwidth, and it can be achieved without extra optical hardware.

3.2.3 QoS algorithm

Initially, the quota for all the nodes (Q_i^0) is set to the size of the epoch T to guarantee high resource utilization. In each epoch t, with the global service information of the past epoch (A_i^{t-1} and b_i^{t-1}) collected from all the nodes, an algorithm is implemented to update the quota for the next epoch (Q_i^{t+1}) for each node i to achieve the weighted max-min fairness. We derive Q_i^{t+1} based on two components: the *base quota* (B_i^{t+1}) and the *adjustment quota* (X_i^{t+1}), as shown in Equation 1. The *base quota* represents the target quota for a node in a stable state, where the low demand nodes achieve their full demand, and the high demand nodes share the remaining resource according to their weights W_i. The *adjustment quota*, on the other hand, tries to punish or compensate a node if it over-consumed or under-consumed service in the past.

$$Q_i^{t+1} = B_i^{t+1} + X_i^{t+1} \tag{1}$$

To calculate the baseline quota (B_i^{t+1}), we first identify *high demand nodes* ($h_i^{t+1} = 1$) by comparing the achieved service level of each node against the average. In epoch t, the QoS controller accumulates A_i^{t-1} to get the *normalized accumulated achieved service* (C_i^{t-1}) for node i up until the $(t-1)$-th epoch. That is,

$$C_i^{t-1} = \frac{\sum_{j=0}^{t-1} A_i^j}{W_i} = C_i^{t-2} + \frac{A_i^{t-1}}{W_i}. \tag{2}$$

Then, the *average normalized accumulated service* ($\hat{C}^{t-1}$) among the *busy* nodes are calculated using the following equation.

$$\hat{C}^{t-1} = \frac{\sum_{i=0}^{K-1} b_i^{t-1} C_i^{t-1}}{\sum_{i=0}^{K-1} b_i^{t-1}} \tag{3}$$

We define that a *high demand node* for the coming epoch ($h_i^{t+1} = 1$) is one that either was busy in the previous epoch ($b_i^{t-1} = 1$) *or* achieved more normalized service than the average.

$$h_i^{t+1} = b_i^{t-1} \vee (C_i^{t-1} \geq \hat{C}^{t-1}) \tag{4}$$

We approximate the total amount of resource that should be shared among the high demand nodes in the coming epoch (S^{t+1}) as the total resource in the epoch (T) subtracted by those achieved by the low-demand nodes

$$S^{t+1} = \alpha \left[T - \sum_{i=0}^{K-1} (1 - h_i^{t+1}) A_i^{t-1} \right] \tag{5}$$

where α is a coefficient that account for the amount of resource that cannot be claimed (i.e., reserved data channel slots for QoS information exchange, tokens that went unused, etc.). Throughout our experiments, we use $\alpha = 0.95$ for good resource utilization.

Thus, the base quota is set based on whether the node is a high demand node: high demand nodes fairly share S^{t+1} based on their weights, and low demand nodes get unlimited quota ($B_i^{t+1} = T$) to let them achieve any amount of bandwidth they need. If there is no busy node, we assert $B_i^{t+1} = T$, otherwise,

$$B_i^{t+1} = (1 - h_i^{t+1})T + h_i^{t+1} \frac{b_i^{t-1} W_i}{\sum_{i=0}^{K-1} b_i^{t-1} W_i} \times S^{t+1} \tag{6}$$

Equation 6 represents the distribution of service among the high-demand nodes. However, this ignores the history effect – nodes that achieved more service in the past are not punished. Such punishment is reflected in the *adjustment quota* X_i^{t+1}, as shown in Equation 7.

As shown above, all the nodes that have achieved C_i^{t-1} higher than $\hat{C}^{t-1}$ are punished by an amount proportional to the excess, while the under-achieving nodes have their quota proportionally relaxed. A slight difference here is that the punishment for over-achieving nodes is more conservative, while the compensation for under-achieving nodes is more aggressive. We normalize the punishment against $\hat{C}^{t-1}$ and a strength coefficient β ($\beta = 0.25$ in all our simulations to avoid fluctuation) is introduced to control how much punishment is given; while the compensation is directly proportional to the difference in the accumulated normalized service, because we do not want to sacrifice resource utilization rate by over-punishing nodes, and it is always safe to compensate the under-achieving nodes. The MAX and MIN functions make sure the total quota Q_i^{t+1}, when calculated using Equation 1, is between 0 and T.

To avoid excessive history effect (i.e., nodes that consumed a large amount of bandwidth a *long* time ago may be punished in current arbitration), and limit the counter size for the service statistics (e.g., C_i^t), we periodically reset theses counters every F cycles. In our evaluations, we use $F = 50,000$ cycles. QoS statistics are thus guaranteed within this time frame F.

3.2.4 Algorithm Overhead

The calculation involved in the algorithm is narrow-width (e.g., less than 16-bit), simple, and scalable. Thus, multiple calculations can fit into a single clock cycle. Data parallelism can also be exploited at the cost of slightly increased hardware cost. More importantly, such calculation is not on the critical path of optical arbitration, and the calculations are carried out only once every epoch of hundreds to thousands of cycles. In drastically scaled technology nodes, the area and power overhead of such electrical circuits will be minimal compared to the less scalable optical hardware and power overheads. For example, we estimate the storage

[3]For example, if Q_i^t is in units of n tokens, m-bit Q_i^t can represent quota of up to $n \times (2^m - 1)$, instead of $2^m - 1$.

$$X_i^{t+1} = \begin{cases} MAX\left(\beta W_i T \dfrac{\hat{C}^{t-1} - C_i^{t-1}}{\hat{C}^{t-1}}, \quad -B_i^{t+1}\right) & \text{if } C_i^t > \hat{C}^{t-1} \\ MIN\left(W_i(\hat{C}^{t-1} - C_i^{t-1}), \quad T - B_i^{t+1}\right) & \text{if } C_i^t \leq \hat{C}^{t-1} \end{cases} \tag{7}$$

Parameter	Value	Parameter	Value
Network size	16,64	Input buffer depth	8 flits
Max concurrent token requests	8	Max transmissions per node per cycle	2
Packet size	64 bytes	Flit size	64 bytes
Clock frequency	5 GHz	Optical link bitrate	10 Gbps/link

Table 3: Simulation Framework Parameters

overhead for T=512, K=64 to be 9.4KB (3% of the total buffers), while the electrical power of all the QoS controllers is 43mW for 64 controllers in 16nm technology, which is $<$ 1% of the power of the best alternative optical arbitration scheme with conservative optical parameters.

3.3 Applying QoS to MWSR Optical Crossbar

FeatherWeight can be used for global arbitration of any resource. For example, in an MWMR crossbar, both credit and channel resources can be arbitrated using FeatherWeight. For simplicity, in this paper, we evaluate it on the channel arbitration of a MWSR crossbar. The receiver channels are thus arbitrated separately, each using a separate set of FeatherWeight optical arbitration, with the QoS controller embedded in the router that the arbitrated channel is dedicated to.

As mentioned before, the QoS information exchange is carried out on the data bus. This is especially feasible because the QoS information exchange has a fixed traffic pattern. In Phase 1 of each epoch (Figure 3(b)), the controllers (at each node) receives the service/busy information from all the other nodes; while in Phase 3, each node receives the updated quota information from all the controllers. As there are multiple such small pieces of information going to a common destination, we can reserve several data slots on the data channel by not injecting the corresponding optical tokens and statically assign which node should use which sub-words in these data slots for QoS information exchange. Note that each data slot is wide enough (e.g., 512-bit) to accommodate the QoS information (A_i^t, b_i^t, and Q_i^t) for many nodes. Thus, the QoS information exchange can be done within very few data slots by car-pooling in limited number of slots on the wide datapath. In our evaluation, we conservatively reserve 4 data slots per epoch on the data channels for QoS information exchange. With this configuration, even considering an 8-cycle light traversal time across all nodes, the QoS information exchange can be done within 10 cycles in each phase of an epoch, which leaves ample time for the algorithm computation in Phase 2.

4. EVALUATION

4.1 System Setup

To evaluate FeatherWeight, a cycle accurate network simulator is developed based on the booksim simulator [3, 4] and modified to model an MWSR nanophotonic crossbar with four different arbitration schemes: baseline optical arbitration, 2-pass Token Stream, Fair Slot, and FeatherWeight. We assume a refractive index of $n = 3.5$ for the waveguide and 1-cycle latency for processing an optical token request [30]. The clock frequency is targeted at 5 GHz. We simulate crossbar sizes of both 16 and 64 ($K = 16, 64$), with single flit packets of 64 bytes. All the schemes employ an input buffer size of 8 flits, implemented as a DAMQ [29], with packets queued separately according to their destination. Each router can initiate a maximum of 8 token requests per cycle, but can utilize at most 2 acquired tokens in each cycle, in accordance with Vantrease et al. [30]. A list of system parameters is provided in Table 3.

4.2 Throughput

We first evaluate the throughput of the different arbitration schemes. As can be seen in Figure 4(a,c), under Hotspot traffic, where all the nodes send to the same destination (Node 0), all the arbitration schemes achieve near-ideal throughput of $1/K$, as they all try to fully utilize the links by arbitrating at time-slot level. However, Fair Slot incurs higher latency before the network reaches saturation: when the network approaches saturation, the wait-time based "hungry" detection in Fair Slot triggers famine mode more easily, and forces all the hungry nodes to take turns to flush their pending packets, which results in higher average latency. With uniform random traffic, Fair Slot loses around 17% of throughput as compared to the other 3 schemes. This is because, at the end of each famine mode, all the nodes are suspended. Thus, the 8 tokens that are on-the-fly in the waveguide cannot be utilized by any node. This incurs a fixed loss of 8 data slots per famine mode. Such cost is amortized in the hotspot traffic where the famine mode is long, but shows up in uniform random traffic, where the sporadic famine modes triggered near network saturation are much shorter. Note that these throughput results are consistent with those reported in the original work [30]. On the other hand, FeatherWeight, with an epoch size of 512 cycles, shows minimal throughput cost for imposing the QoS scheme, even with a few data slots dedicated for QoS information exchange.

4.3 Fairness

The throughput results show how the arbitration performs before network saturation and the best-effort baseline optical arbitration scheme naturally shows good performance. However, if the demand for receiver channel further increases, some nodes in the baseline optical arbitration will

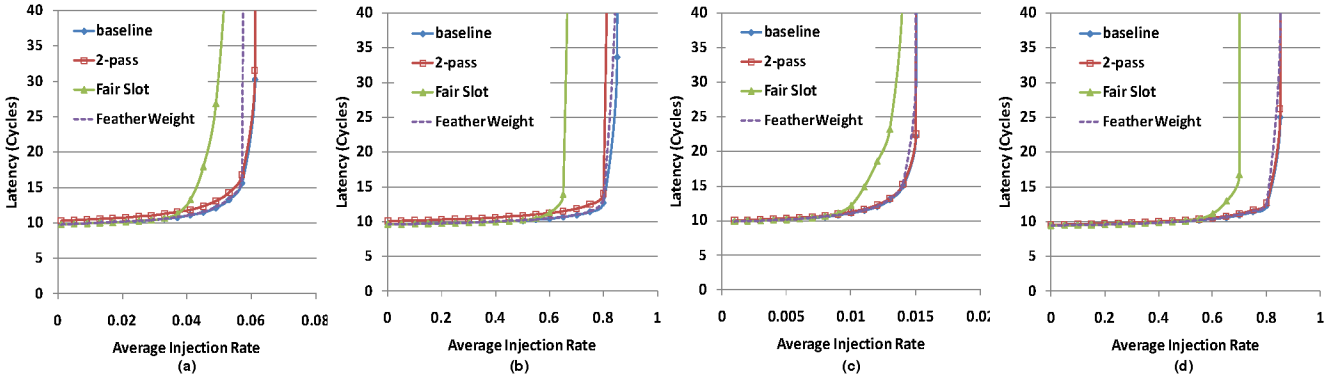

Figure 4: Latency for $K = 16$ (a,b) and 64 (c,d) under Hotspot (a,c) and Uniform (b,d) traffic.

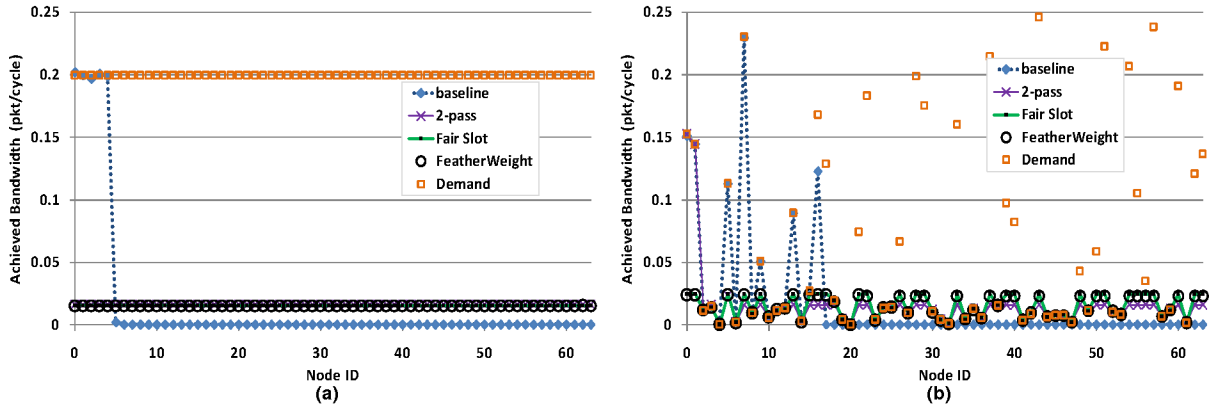

Figure 5: Achieved bandwidth per-node on the channel dedicated to Node 0. (a) equal demand of 0.2 pkt/cycle (b) random demand. Demand levels of each node are indicated by the squares.

be starved, which necessitates the fairness guarantees implemented in the other 3 schemes.

Figure 5 shows the per-node achieved bandwidth under different arbitration schemes when the receiver channel of node 0 is over-subscribed in hotspot traffic. All the nodes are given equal weights in FeatherWeight, and an epoch size of 512 is used. We simulate two demand scenarios: (a) every node has an equal demand of 0.2, and (b) half of the nodes have a random low demand between 0 and 0.01, while the others have a random high demand between 0 and 0.25, as indicated by the boxes in the figure. In both cases, the baseline optical arbitration scheme only satisfies the first few nodes, starving the remaining nodes. In scenario (a), the 2-pass scheme can fairly distribute the bandwidth to all the nodes. But, when the per-node demand is unbalanced as in scenario (b), the unclaimed bandwidth of the low-demand nodes are all consumed by the first few high demand nodes, leaving the remaining nodes with only at most their dedicated bandwidth ($1/K$). In both scenarios, Fair Slot[4] and FeatherWeight achieve perfect max-min fairness, with high-demand nodes enjoying the same level of bandwidth while low demand nodes achieving their respective bandwidth demand. The total resource utilization rate is above 99% for both Fair Slot and FeatherWeight. This demonstrates that FeatherWeight is capable of achieving perfect fairness with minimal optical hardware overhead.

4.4 Differentiated service

By assigning different weights to different nodes, FeatherWeight can alter the amount of resource each node acquires. Figure 6(a) demonstrates the arbitration of a single receiving channel (of Node 0) in FeatherWeight where certain nodes are given higher weights. We evaluate 3 scenarios: "equal", where all nodes have the default weight of 1; "4x4", where nodes 0, 16, 32, and 48 have a weight of 4 and the others have a weight of 1; "linear", where nodes $8i+7(i = 0...7)$ have a weight of $i + 2$, and the others have a weight of 1. In all these cases, FeatherWeight perfectly achieves the weighted max-min fairness, providing differentiated service levels to different nodes according to their weights.

4.5 Performance Isolation

Performance isolation is another important QoS feature that can, for example, control the impact of potential Denial-of-Service (DoS) attacks [20]. In this section, we evaluate a case where 4 attackers are continuously sending packets to a single receiver channel in a 64-node MWSR crossbar. As the attacker location plays an important role in the performance

[4]The Fair Slot scheme [30] implemented here is an improved implementation over its initial publication. Equal bandwidth division among the senders can be achieved after the modification, though the scheme is exactly as described previously.

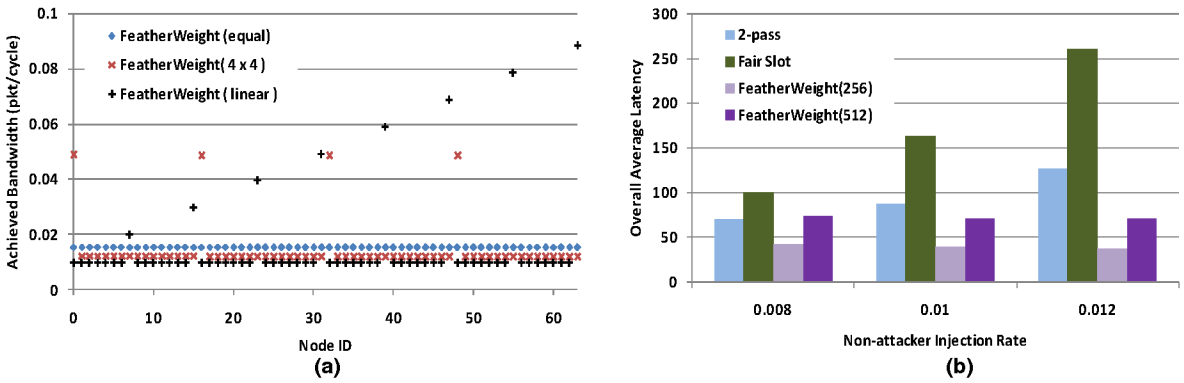

Figure 6: (a) Differentiated service provided by FeatherWeight and (b) average latency of non-attacker nodes under Denial-of-Service attack.

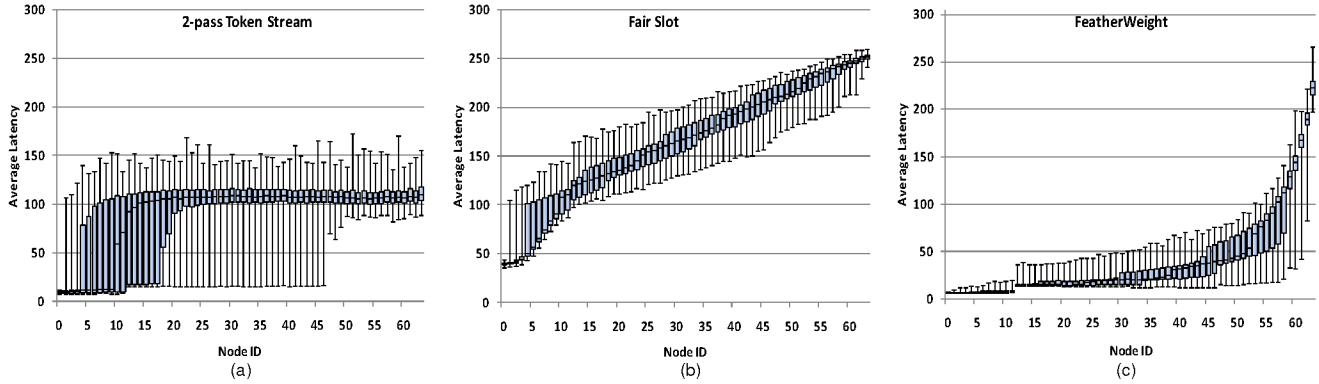

Figure 7: Per-node average latency of non-attackers under Denial-of-Service (DoS) attack for (a) 2-pass Token Stream, (b) Fair Slot, and (c) FeatherWeight with epoch size of 256 cycles.

by all the schemes, a total of 1024 random locations of the 4 attackers are simulated. The attackers all have a non-stop bandwidth demand of 1.0 pkt/cycle, while the non-attacker nodes have the same moderate bandwidth demand between 0.008 pkt/cycle and 0.012 pkt/cycle. All the nodes have the same weight in FeatherWeight.

Figure 6(b) shows the *overall* average latency of all the non-attacker nodes across the 1024 attack scenarios. FeatherWeight schemes with epoch sizes of 256 and 512 are shown. It can be seen that FeatherWeight is insensitive to the non-attacker traffic load, since all the nodes enjoy reserved bandwidth, and the latency is proportional to the epoch size. Fair Slot mostly operates in famine mode with each node sending packets in sequence, hence its latency is proportional to the amount of total traffic within each famine mode. Overall, with a non-attacker injection rate of 0.01 pkt/cycle, FeatherWeight with an epoch size of 256 cycles reduces the non-attacker average latency by 76% and 56% compared to the Fair Slot and 2-pass schemes, respectively.

To better understand the latency distribution with different arbitration schemes, we also summarize the per-node average latency for non-attacker injection rate of 0.01 pkt/cycle, as shown in the whisker-box plots in Figure 7. The whiskers indicate the maximum and minimum latencies, while the boxes indicate the 25%, 50%, and 75% quantiles across the 1024 attacker scenarios. In 2-pass Token Stream arbitration, nodes before the first attacker have abundant bandwidth and hence experience low latency, while the other nodes can only use their dedicated tokens. Thus, the first few nodes have wide latency distribution across the 1024 scenarios, while the majority of nodes have a tight distribution around 100 cycles. In Fair Slot, the latency is linearly increasing according to the nodes' physical location, and it is relatively independent of the attacker location as all the nodes take turns to send packets in the famine mode. FeatherWeight, on the other hand, provides very low latency for most nodes, and only the last few nodes experience higher latency. This is because, any node before the last attacker node will have abundant bandwidth, and thus its latency will be low. The last few nodes experience higher latency that is proportional to the epoch size. Note that in this figure, latencies of each node comes from different attacker scenarios, so it is always better to have lower latency rather than keeping a tight latency distribution per node.

4.6 Trace Traffic

Limited by prohibitive length of full system simulations, we evaluate FeatherWeight with traces extracted from full system simulation with the GEMS [19]/ SIMICS [18] framework. We use 9 traces from MineBench [21] and SPLASH-2 [33] benchmarks. To stress the network, we only use the network requests, and let the node with the most network requests inject requests at 1.0 pkt/cycle. All the other nodes generate requests at a rate proportional to its total number

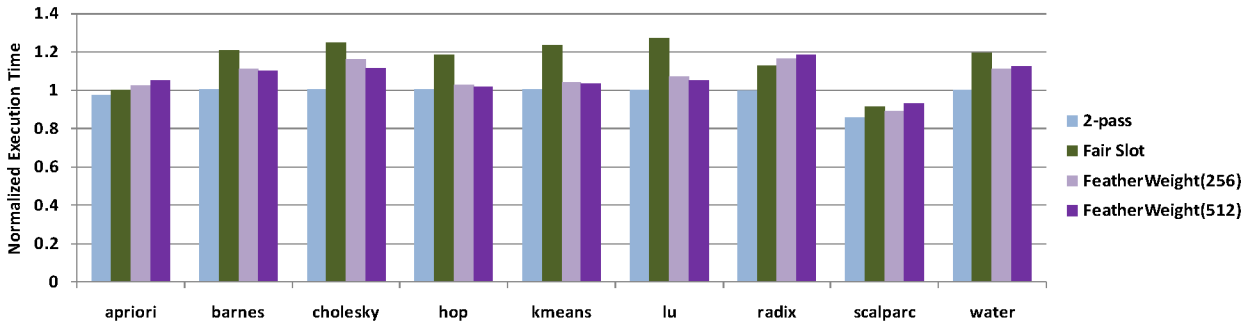

Figure 8: Execution time of traces from SPLASH-2 and MineBench applications normalized to baseline optical abitration scheme

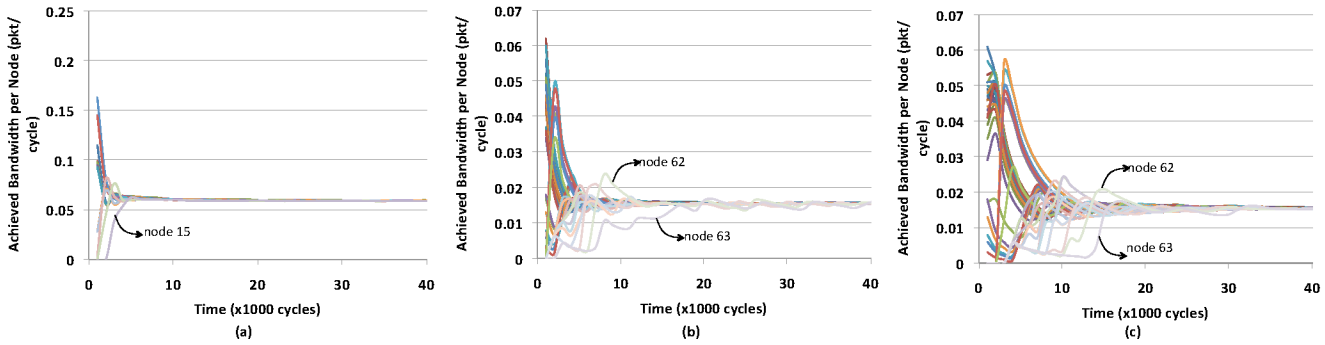

Figure 9: Time response of achieved bandwidth for FeatherWeight with (a) $K = 16, T = 256$, (b) $K = 64, T = 512$, and (c) $K = 64, T = 1024$.

of requests. Any node that receives a request will automatically generate a reply packet, and the reply packets have priority in entering the network over requests. Each node can have a maximum of 16 outstanding requests, beyond which the node is blocked from injecting more request packets. In this case, the time-variant contention on each channel is in general changing faster than a real-world traffic load, and hence it serves as an adversarial test case for FeatherWeight. The execution time is normalized to the baseline optical arbitration scheme, as shown in Figure 8.

For *cholesky*, the network contention is low, and it is advantageous to avoid unnecessarily triggering the QoS management, as the throttling may limit channel utilization. In such cases, having longer epoch makes it less likely for throttling to be applied, and the overall performance of FeatherWeight(512) is better. As the Fair Slot scheme has a more stringent criteria for "hungry" node detection (where a wait counter is used for each packet), it tends to enter the famine mode more easily and incurs more throughput loss. However, there are also cases like *scalparc* and *apriori*, where the trace poses more time-variant contention. In these cases, it is better to have a shorter epoch to closely track the time variant traffic pattern.

On average, FeatherWeight reduces the trace execution time by 7.5% as compared to Fair Slot and incurs performance overhead of 7.0% and 9.0% as compared to the baseline and the 2-pass Token Stream, respectively. Note that these two schemes do not guarantee fairness.

4.7 Time Response

It is important to understand how fast our feedback controlled QoS system responds to change in traffic patterns. Here we analyze the arbitration of a single channel and assign a full quota of T to each node initially. A total bandwidth demand of 3.2× the available bandwidth is equally distributed to all the nodes to create abundant contention. Figure 9 shows the achieved bandwidth per node as time progresses. Each line represents a different node. Clearly, nodes closer to *home* initially achieve much higher bandwidth than the nodes farther away. As time progresses, the feedback control system throttles the over-achievers and compensates the under-achievers. All the nodes quickly converge to the same bandwidth consumption. The initially starving nodes are satisfied in sequence according to their physical location, and the farthest node is the last to achieve its fair share of bandwidth. Systems with fewer requesting nodes are quicker in converging, and shorter epochs also help the system to quickly achieve fairness. Even with 64 busy nodes and an epoch size of 1024 cycles, FeatherWeight achieves fairness within 30,000 cycles, which is far shorter than a typical program phase. For the 16-node case and an epoch size if 256 cycles, fairness can be achieved within 5,000 cycles.

4.8 Power Cost

As an emerging technology, many parameters of the nanophotonic devices are subject to change as the technology progresses. We adopt the optical power model by Josh et al. [10] and apply two sets of technology parameters [6, 10, 30, 37] as

Parameter	Conservative	Aggressive	Parameter	Conservative	Aggressive
Waveguide loss	1 dB/cm	0.4 dB/cm	Modulator insertion	1e-2 dB/ring	1e-3 dB/ring
Through-ring loss	1e-3 dB/ring	5e-4 dB/ring	Coupler/splitter	2 dB	1dB
Filter drop	3 dB	1.5 dB	Detector sensitivity	3 uW	1.5 uW
Modulation	80 pJ/bit	20 pJ/bit	Demodulation	40 pJ/bit	20 pJ/bit
Ring heating	40 uW/ring	15 uW/ring			

Table 4: Nanophotonic Device Power Parameters

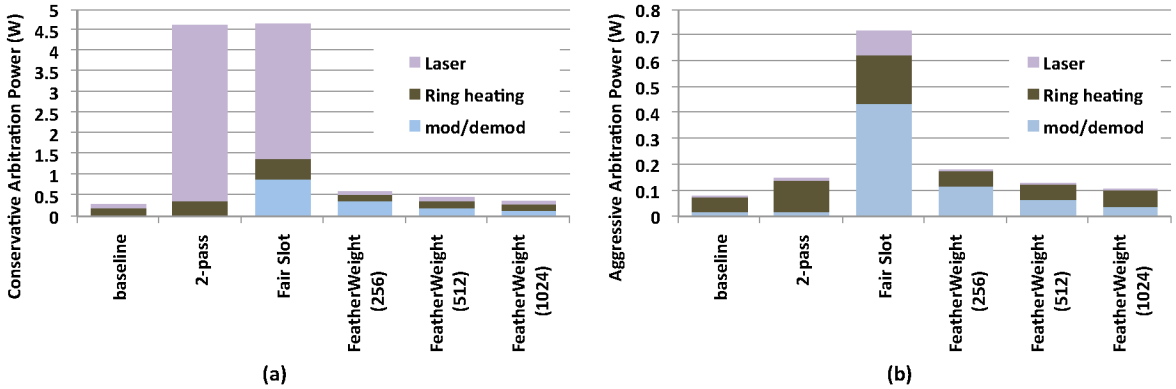

Figure 10: Power consumption under (a) conservative and (b) aggressive device parameters

listed in Table 4. Aggressive parameters represent the more optimistic/challenging projection regarding the technology than conservative parameters.

Figure 10(a) shows the power consumption under conservative technology parameters. Due to the lossy waveguide and less sensitive detectors, optical power is dominant in 2-pass Token Stream and Fair Slot, because of their long waveguide length and the broadcast bus, respectively. The larger number of ring resonators in these schemes also incurs high ring heating power. However, the overhead for implementing QoS in FeatherWeight only lies in the periodically exchanging QoS information on the data bus, which is amortized by the epoch size. Thus, FeatherWeight with an epoch size of 256 reduces total power consumption by 87%, as compared to Fair Slot, while providing the support for differentiated service. Compared to the baseline optical arbitration scheme, FeatherWeight consumes no more than 2× the energy while providing not only freedom from starvation, but also fairness, performance isolation, and differentiated service. The total power consumption of less than 0.7W makes FeatherWeight easily feasible for most CMP power budget designs.

With the aggressive technology parameters, FeatherWeight is still most efficient as it has minimal optical hardware, and the information exchange on the data bus is limited. In this case, FeatherWeight with epoch size of 256 achieves 75% power reduction compared to Fair Slot, and incurs only 1.3× overhead compared to the baseline scheme.

5. CONCLUSION

In this paper, we presented FeatherWeight, a lightweight optical arbitration scheme that significantly reduces the cost of optical arbitration compared to prior art. This can enable the applicability of global optical arbitration in power-bounded future many-core processors. In addition to lower cost, features of nanophotonics is leveraged to efficiently provide QoS support in the global arbitration. We demonstrated that FeatherWeight achieves 17% higher throughput, 7.5% shorter trace execution time while reducing the power consumption by up to 86%, when compared to the best alternative.

Acknowledgements

This work is supported by NSF grants CCF-0916746, CNS-0750847, CCF-0747201, and CCF-0720691. John Kim was supported by WCU (World Class University) program under the National Research Foundation of Korea and funded by the Ministry of Eduation, Science and Technology of Korea (Project No: R31-30007). We would like to also thank all the anonymous referees for their detailed comments.

6. REFERENCES

[1] AMD Fusion Family APUs. `http://sites.amd.com/us/fusion/apu/Pages/fusion.aspx`.

[2] J. Ahn, M. Fiorentino, R. Beausoleil, N. Binkert, A. Davis, D. Fattal, N. Jouppi, M. McLaren, C. Santori, R. Schreiber, et al. Devices and architectures for photonic chip-scale integration. *Applied Physics A: Materials Science & Processing*, 95(4):989–997, 2009.

[3] J. Balfour and W. J. Dally. Design tradeoffs for tiled CMP on-chip networks. In *Proc. of the Int'l Conference on Supercomputing (ICS)*, pages 187–198, Carns, Queensland, Australia, 2006.

[4] W. J. Dally and T. B. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishing Inc., 2004.

[5] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. In *Symposium proceedings on Communications architectures & protocols*, pages 1–12. ACM, 1989.

[6] R. K. Dokania and A. B. Apsel. Analysis of challenges for on-chip optical interconnects. In *GLSVLSI '09: Proceedings of the 19th ACM Great Lakes symposium on VLSI*, pages 275–280, New York, NY, USA, 2009. ACM.

[7] B. Grot, S. W. Keckler, and O. Mutlu. Preemptive virtual clock: A flexible, efficient, and cost-effective qos scheme for networks-on-chip. In *IEEE/ACM Int'l Symposium on Microarchitecture (MICRO)*, pages 163–174, New York, NY, 2009.

[8] IBM. Photonics `www.research.ibm.com/photonics/`.

[9] Intel. Photonics `http://techresearch.intel.com/ResearchAreaDetails.aspx?Id=26/`.

[10] A. Joshi, C. Batten, Y.-J. Kwon, S. Beamer, I. Shamim, K. Asanovic, and V. Stojanovic. Silicon-photonic clos networks for global on-chip communication. In *IEEE Int'l Symposium on Network-on-Chip (NOCS)*, San Diego, CA, 2009.

[11] S. Keshav. *An Engineering Approach to Computer Networking: ATM Networks, The Internet and Telephone Network*, pages 215–217. Addison-Wesley Professional, 1997.

[12] J. Kim, W. J. Dally, B. Towles, and A. Gupta. Microarchitecture of a high-radix router. In *Proc. of the Int'l Symposium on Computer Architecture (ISCA)*, Madison, WI, Jun. 2005.

[13] N. Kirman, M. Kirman, R. K. Dokania, J. F. Martinez, A. B. Apsel, M. A. Watkins, and D. H. Albonesi. Leveraging optical technology in future bus-based chip multiprocessors. In *IEEE/ACM Int'l Symposium on Microarchitecture (MICRO)*, pages 492–503, Orlando, FL, 2006.

[14] N. Kirman and J. Martínez. A power-efficient all-optical on-chip interconnect using wavelength-based oblivious routing. In *Proceedings of the fifteenth edition of ASPLOS on Architectural support for programming languages and operating systems*, pages 15–28. ACM, 2010.

[15] P. Koka, M. O. McCracken, H. Schwetman, X. Zheng, R. Ho, and A. V. Krishnamoorthy. Silicon-photonic network architectures for scalable, power-efficient multi-chip systems. In *Proc. of the Int'l Symposium on Computer Architecture (ISCA)*, pages 117–128, New York, NY, USA, 2010. ACM.

[16] S. Kottapalli and J. Baxter. Nehalem-EX CPU Architecture. *Hot Chips (Aug 2009)*.

[17] J. Lee, M. Ng, and K. Asanovic. Globally-synchronized frames for guaranteed quality-of-service in on-chip networks. In *Proc. of the Int'l Symposium on Computer Architecture (ISCA)*, pages 89–100, Beijing, China, 2008.

[18] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hallberg, J. Hogberg, F. Larsson, A. Moestedt, and B. Werner. Simics: A full system simulation platform. *Computer*, 35(2):50–58, Feb. 2002.

[19] M. M. Martin, D. J. Sorin, B. M. Beckmann, M. R. Marty, M. Xu, A. R. Alameldeen, K. E. Moore, M. D. Hill, and D. A. Wood. Multifacet's general execution-driven multiprocessor simulator (gems) toolset. *ACM SIGARCH Computer Architecture News*, 33(4):92–99, Sep. 2005.

[20] T. Moscibroda and O. Mutlu. Memory performance attacks: Denial of memory service in multi-core systems. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, pages 1–18. USENIX Association, 2007.

[21] R. Narayanan, B. Ozisikyilmaz, J. Zambreno, G. Memik, and A. Choudhary. Minebench: A benchmark suite for data mining workloads. In *IEEE Int'l Symposium on Workload Characterization (IISCW)*, pages 182–188, San Jose, CA, 2006.

[22] K. Nesbit, N. Aggarwal, J. Laudon, and J. Smith. Fair queuing memory systems. In *IEEE/ACM Int'l Symposium on Microarchitecture (MICRO)*, pages 208–222, Orlando, FL, 2006.

[23] K. Nesbit, J. Laudon, and J. Smith. Virtual private caches. In *Proc. of the Int'l Symposium on Computer Architecture (ISCA)*, page 68, San Diego, CA, 2007. ACM.

[24] J. Ouyang and Y. Xie. LOFT: A High Performance Network-on-Chip Providing Quality-of-Service Support. In *Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture*, pages 409–420. IEEE Computer Society, 2010.

[25] J. Ouyang and Y. Xie. Enabling quality-of-service in nanophotonic network-on-chip. In *Proceedings of the 16th Asia and South Pacific Design Automation Conference*, pages 351–356. IEEE Press, 2011.

[26] Y. Pan, J. Kim, and G. Memik. Flexishare: Channel sharing for an energy-efficient nanophotonic crossbar. In *Int'l Symposium on High-Performance Computer Architecture (HPCA)*, pages 1–12, Bangalore, India, Jan. 2010.

[27] Y. Pan, P. Kumar, J. Kim, G. Memik, Y. Zhang, and A. Choudhary. Firefly: Illuminating future network-on-chip with nanophotonics. In *Proc. of the Int'l Symposium on Computer Architecture (ISCA)*, Austin, TX, 2009.

[28] A. Shacham, K. Bergman, and L. Carloni. Photonic networks-on-chip for future generations of chip multiprocessors. *Computers, IEEE Transactions on*, 57(9):1246–1260, 2008.

[29] Y. Tamir and G. Frazier. Dynamically-allocated multi-queue buffers for VLSI communication switches. *IEEE Transactions on Computers*, pages 725–737, 1992.

[30] D. Vantrease, N. L. Binkert, R. Schreiber, and M. H. Lipasti. Light speed arbitration and flow control for nanophotonic interconnects. In *IEEE/ACM Int'l Symposium on Microarchitecture (MICRO)*, pages 304–315, New York, NY, 2009.

[31] D. Vantrease, R. Schreiber, M. Monchiero, M. McLaren, N. P. Jouppi, M. Fiorentino, A. Davis, N. L. Binkert, R. G. Beausoleil, and J. H. Ahn. Corona: System implications of emerging nanophotonic technology. In *Proc. of the Int'l Symposium on Computer Architecture (ISCA)*, pages 153–164, Beijing, China, 2008.

[32] Z. Wang. *Internet QoS: architectures and mechanisms for quality of service*. Morgan Kaufmann Publishers, 2001.

[33] S. Woo, M. Ohara, E. Torrie, J. Singh, and A. Gupta. The SPLASH-2 programs: Characterization and methodological considerations. In *Proc. of the Int'l Symposium on Computer Architecture (ISCA)*, pages 24–36, Santa Margherita Ligure, Italy, Jun. 1995.

[34] Q. Xu, S. Manipatruni, B. Schmidt, J. Shakya, and M. Lipson. 12.5 gbit/s carrier-injection-based silicon micro-ring silicon modulators. *Opt. Express*, 15(2):430–436, Jan. 2007.

[35] T. Yin, R. Cohen, M. M. Morse, G. Sarid, Y. Chetrit, D. Rubin, and M. J. Paniccia. 40gb/s ge-on-soi waveguide photodetectors by selective ge growth. In *Conference on Optical Fiber communication/National Fiber Optic Engineers Conference (OFC/NFOEC)*, pages 24–28, San Diego, CA, 2008.

[36] L. Zhang. VirtualClock: a new traffic control algorithm for packet-switched networks. *ACM Transactions on Computer Systems (TOCS)*, 9(2):124, 1991.

[37] L. Zheng, A. Mickelson, L. Shang, M. Vachharajani, D. Filipovic, W. Park, and Y. Sun. Spectrum: A hybrid nanophotonic-electric on-chip network. In *Proc. of Design Automation Conference (DAC)*, San Francisco, CA, Jun 2009.