Probabilistic Distance-based Arbitration: Providing Equality of Service for Many-core CMPs

Michael M. Lee, John Kim
KAIST
{minjong7, jjk12}@kaist.edu

Dennis Abts, Michael Marty
Google Inc.
{dabts, mikemarty}@google.com

Jae W. Lee
Parakinetics Inc.
leejw@parakinetics.com

***Abstract*—Emerging many-core chip multiprocessors will integrate dozens of small processing cores with an on-chip interconnect consisting of point-to-point links. The interconnect enables the processing cores to not only communicate, but to share common resources such as main memory resources and I/O controllers. In this work, we propose an arbitration scheme to enable equality of service (EoS) in access to a chip's shared resources. That is, we seek to remove any bias in a core's access to a shared resource based on its location in the CMP.**

We propose using *probabilistic* arbitration combined with distance-based weights to achieve EoS and overcome the limitation of conventional round-robin arbiter. We describe how nonlinear weights need to be used with probabilistic arbiters and propose three different arbitration weight metrics – fixed weight, constantly increasing weight, and variably increasing weight. By only modifying the arbitration of an on-chip router, we do not require any additional buffers or virtual channels and create a simple, low-cost mechanism for achieving EoS. We evaluate our arbitration scheme across a wide range of traffic patterns. In addition to providing EoS, the proposed arbitration has additional benefits which include providing quality-of-service features (such as differentiated service) and providing fairness in terms of both throughput and latency that approaches the global fairness achieved with age-base arbitration – thus, providing a more stable network by achieving high sustained throughput beyond saturation.

***Keywords*- on-chip network; age-based arbitration; fairness; quality of service (QoS);**

I. Introduction

Emerging many-core chip multiprocessors will integrate dozens of small processing cores with an on-chip interconnect consisting of point-to-point links. The interconnect enables the processing cores to not only communicate, but to share common resources such as main memory resources and I/O controllers. In particular, accessing memory from shared memory controllers is especially performance sensitive and these types of systems will introduce non-uniformity into memory and I/O access. We find ourselves in the tenuous design space of being capable of implementing many cores, with only a few memory and I/O controllers. This brings new light to old problems of providing *equality of service* to a set of shared resources regardless of where, or which processing core, is scheduled to execute a thread.

The on-chip network is crucial to providing equality of service among the shared resources; providing consistent latency and bandwidth characteristics regardless of the origin or destination of the communication. The goal is to provide bandwidth and latency characteristics that are consistent for all processors on chip. Applications should be insensitive to *where*, within the on-chip network, the thread is scheduled to execute. Achieving this goal will reduce the variance in the execution time among threads and will provide more efficient synchronization when transitioning between parallel and sequential code regions. For instance, we seek to prevent a core from receiving unfair and unequal bias to a neighboring memory controller resource, compared with a core located further away. Thus, *equality of service* (EoS) provides equal access to shared network resources regardless of location.

Recently, cost-efficient quality of service (QoS) for on-chip networks have been proposed [1], [2]. Unlike QoS, which strives to provide differentiated service and hard (or soft) guarantees for end-to-end latency or bandwidth profile, EoS does not provide guarantees yet provides equal access to shared on-chip resources. [1] In this work, we propose and evaluate an arbitration mechanism to achieve equality of service and predictable performance. By tackling arbitration in the interconnect, we ensure the packets delivered to a shared CMP resource are not unfairly biased by source location. However, for *on*-chip networks, arbitration must be fast and simple to reduce overheads.

In this paper, we introduce *distance-based* arbitration by taking into account the distance or the hop count which a packet travels en route to its destination – allowing nodes located many hops from the edge to get equal service compared to a node close to the edge. We propose using *probabilistic* arbitration with a distance-based selection algorithm to achieve EoS while providing a low complexity, livelock-free arbitration allowing for consistent latency and bandwidth characteristics for all cores. Since nodes that are farther away are serviced at a ratio that is geometrically proportional to the hop count, we propose using *nonlinear* weights in probabilistic arbitration to provide fairness to nodes that are farther away. Three different arbitration weight metrics are proposed which all provide EoS but have varying trade-off in terms of complexity and performance degradation on different traffic patterns.

Specifically, the contributions of this work include the following:

- We introduce distance-based arbitration as a metric to approximate ages with hop count.

[1] EoS can be considered a subset of QoS.

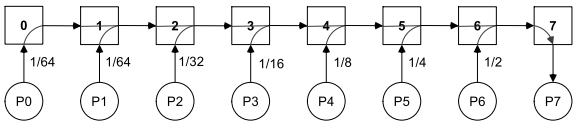


Fig. 1: 8-ary 1-mesh example where all nodes are sending to P7 and merging traffic at each hop.

- To provide fairness using priority-based arbitration [3] with distance as a metric, we propose a distributed *probabilistic* arbitration where arbitration decisions are made probabilistically at each router based on the weights of input requests.
- We propose how nonlinear weights are required to achieve equality-of-service (EoS) and describe three different arbitration weight metrics based on the hop count and the degree of contention.
- We show how distance-based, probabilistic arbitration can provide additional benefits which include providing QoS-like characteristics and provide stronger fairness than conventional round-robin arbitration to enable a more stable network.

II. Motivation

A. Need for Equality of Service

There are many different examples of when equality of service (EoS) is needed in future CMP processors. EoS is crucial to achieve good utilization of hardware resources with multi-threaded programs under non-uniform access time to a critical shared resource since execution time of the slowest thread determines the overall performance. With EoS, the need for topology-aware mapping of threads and tasks reduces as equality of service can be achieved regardless of placement. For hot-spot traffic that occur such as with core-memory traffic [4] where fewer number of memory controllers compared to cores create hotspot traffic around memory controllers and EoS is required for this traffic as well. In this work, we describe how probabilistic arbitration with the proposed distance-based metrics can provide this EoS without the complexity of supporting age-based arbitration.

B. EoS problem in On-Chip Network

As traffic flows through the network, it merges with newly injected packets and traffic from other directions in the network. This merging of traffic from different sources causes packets that have further to travel (more hops) to receive geometrically less bandwidth. For example, consider the 8-ary 1-mesh in Figure 1 where processors P0 thru P6 are sending to P7. The switch allocates the output port by granting packets fairly among the input ports. With a round-robin arbitration policy, the processor closest to the destination (P6 is only one hop away) will get the most bandwidth — 1/2 of the available bandwidth. The processor two hops away, P5, will get half of the bandwidth into router R6, for a total of 1/2×1/2 = 1/4 of the available bandwidth. That is, every two arbitration cycles P7 will deliver a packet from source P6, and every four arbitration cycles it will deliver a packet from source P5. As a result, P0 and P1 each receive only 1/64 of the available bandwidth into P7, a factor of 32 times less than that of P6. Reducing the variation in bandwidth is critical for application performance, particularly as applications are scaled to higher processor counts. Although round-robin arbitration provides local fairness at each router, it does not provide any global fairness across all routers. Age-based arbitration [5] is known to provide global fairness as when two or more packets arbitrate for a shared resource, the packet with the oldest age wins the arbitration. However, it become complex to implement in an on-chip network constraint.

In this work, we avoid the complexity with age-based arbitration by proposing to approximate the age of a packet with *distance* or hop count. By using information already present in the packet, such as source node, current node, or destination node and using distance as a proxy for the packet's age, age-based arbitration is greatly simplified. To understand how hop count can approximate age, the age of a packet corresponds to the latency (T) of a packet from the source node to its destination.

$$\begin{aligned} T &= T_h + T_s + T_w + T_c \\ &= Ht_r + T_s + Ht_w + Ht_q \\ &= H(t_r + t_w + t_q) + T_s \end{aligned}$$

where T_h is the header latency, T_s is the serialization latency, T_w is the wire delay, and T_c is the contention and queuing latency. For all packets, T_s is constant, regardless of the total latency and is only dependent on the channel bandwidth and packet size. For all the other parameters, they are directly proportional to the total hop count (H) from source to destination and other parameters such as per-hop router latency (t_r), per-hop wire delay (t_w), and per-hop queuing delay (t_q). Since we assume a 2D mesh topology, all t_w are identical. We approximate T_c with Ht_q since we assume per-hop queuing latency dominates the contention latency. Thus, the age of a packet is directly proportional to the hop count (H) and can be used to approximate the packet's age. In this paper, we show how the hop count can be used with probabilistic arbitration and nonlinear weight priorities to provide equality of service.

III. Arbitration Design

In order to use hop count as an arbitration metric, we need to guarantee fairness since by providing preference based on weights, there is potential for livelock and starvation. In this section, we propose probabilistic arbitration which can provide fairness while using hop count to determine the weight of packets. We also present how *nonlinear* weights are required using hop count and different weight metrics are presented.

A. Probabilistic Arbitration

Previous arbitration architectures are *deterministic*– that is, given a set of input requests and the switch's current state (such as a state of the arbitration pointer or priorities), the output grants are always deterministically assigned. For sorted priority-based arbitration [3] such as age-based arbitration, arbitration is done deterministically based on the relative age of the requests. Starvation is inherently not a problem with age-based arbitration. By using priority based on hop

Fig. 2: High-level block diagram of a probabilistic arbiter.

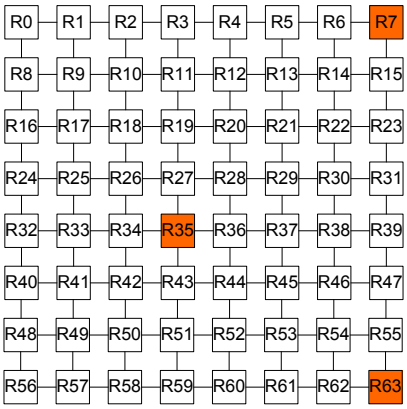


Fig. 3: 8x8 2D-mesh block diagram with several hotspots highlighted. The details of hotspot traffic to these nodes are shown in Figure 4.

count, livelock and fairness issues are problematic because packets with a lower priority (i.e., a lower hop count) can continually lose arbitration because of a constant stream of newly injected traffic with higher priority. To overcome this problem while still using hop count as the weight, we propose *probabilistic* arbitration where the output of the arbitration is probabilistically determined based on the weight of the input requests.

Assume an arbiter shown in Figure 2 with two request r_1 and r_2, each with a corresponding weight w_1 and w_2. The probability of each grant g_1 and g_2 being asserted with probabilistic arbitration is equal to the following.

$$P(g_1) = \frac{w_1}{w_1 + w_2}$$
$$P(g_2) = \frac{w_2}{w_1 + w_2}$$

Since both grants cannot be asserted in the same cycle, the arbiter needs to probabilistically select one of the two requests based on this probability. If the incoming weights are identical (i.e. $w_1 = w_2$), the arbiter behaves like a random arbiter – randomly selecting one of the two requests. In general, for a request r_i into an arbiter with m requests, the probability of r_i being granted is

$$P(g_i) = \frac{w_i}{\sum_{j=1}^{j=m} w_j}$$

A request in probabilistic arbitration will not starve indefinitely, since with probability one, it will be granted regardless of the weight. However a request can incur significant wait time if it continuously loses arbitration. Further discussion of starvation is discussed in Section VI-A.

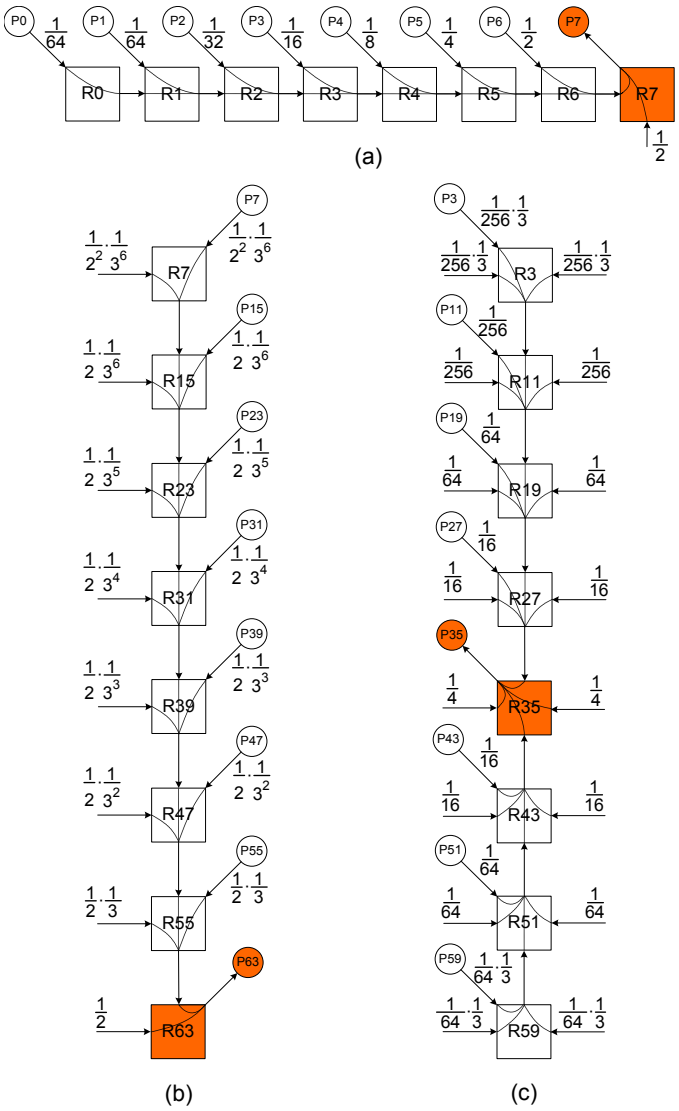


Fig. 4: Merging traffic in 8x8 2D-mesh. The fraction numbers represent the amount of bandwidth that the corresponding nodes would be received if a locally fair, round-robin arbitration is implemented. The highlighted node represents hotspot traffic in a 2D mesh network shown in Figure 3.

B. Linear Weight

The hop count can be implemented as *linear* weights in probabilistic arbiter – i.e., $w = h_x$ or $w = h_x + h_y$ where h_x and h_y represent the hop count from source to destination in each dimension. However, probabilistic arbitration using linear weight hop count cannot provide EoS since farther nodes will be serviced linearly instead of geometrically. The weight inputs to the probabilistic arbiter will only differ linearly and not be able to provide EoS to farther nodes. For example, for two packets that are separated by x hop count, the linear weights for the two packets will be w and $w - x$ – assuming both packets have the same destination. The probability of each packet winning an arbitration is $\frac{w}{2w-x}$ and $\frac{w-x}{2w-x}$, respectively. For large values of w ($w \gg x$) or for small values of x, the probability of each packet winning the arbitration is

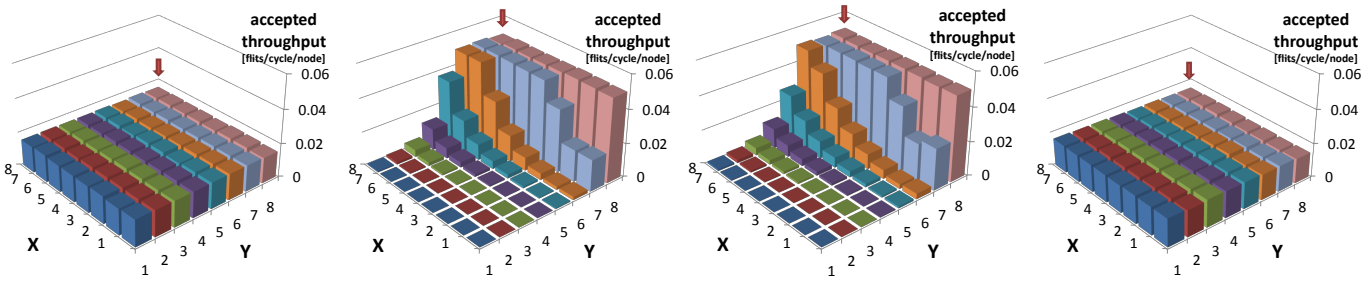


Fig. 5: Accepted throughput per source node by a hotspot resource (indicated by arrow) using (a) age-based arbitration, (b) round-robin arbitration, (c) probabilistic arbitration with linear weights, and (d) probabilistic arbitration with non-linear weights.

approximately $1/2$. Thus, the result of probabilistic arbitration with linear weight is very similar to round-robin arbitration. This is shown with the results in Figure 5(c) for hotspot traffic where all traffic is sent to a single node. [2] The resulting acceptance rate of each node is very similar to the round-robin arbitration shown in Figure 5(b) and does not provide the equality of service as shown in Figure 5(a) with an ideal age-based arbitration.

C. Nonlinear Weight

As shown earlier in Figure 1, nodes that are farther away are serviced at a rate that is exponentially proportional to the hop count – for example, packets that are h hops away are serviced at a rate of $(1/2)^h$ and the service rate is not linearly proportional to the hop count. To account for this difference, we introduce *nonlinear* weights based on the distance. Instead of using a weight which is equal to the hop count (i.e., $w = h$), we introduce nonlinear weights in probabilistic arbitration – i.e., $w = C^h$ where C is the *contention degree* or the number of packets contending for the same output port. By using nonlinear weights, better fairness is provided for nodes that are farther away. For example, in Figure 4(a), if nodes are serviced at a rate of $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, ...$, in order to provide EoS, each node needs to be prioritized with a weight of $2, 4, 8, 16, ...$, respectively. Thus, for the traffic pattern shown in Figure 4(a), $w = 2^h$ can be used with probabilistic arbitration to achieve EoS. $C = 2$ value is used since for each output, there are two flows contending for a router output. With XY routing, packets traveling in the x-dimension will merge similar to the traffic shown in Figure 4(a).

For hotspot traffic shown in Figure 4(b), when traversing the y-dimension, there are 3 traffic flows merging at each router, resulting in each flow being serviced at a rate of $1/3$. Thus, weight used for y-dimension is $w = 3^h$. For traffic shown in Figure 4(c) where the destination node is located in the non-edge location of a 2D mesh network, the number of flows merging is 4, thus $w = 4^h$ needs to be used to provide fairness across all nodes.

D. Arbitration Weight Metrics

To better understand some of the design tradeoffs, we first define several metrics that can be used as an input to probabilistic arbitration. The hop count weight used can be categorized as either *static* or *dynamic*. With a static arbitration metric, the priority of the packet is known beforehand at the time the packet is injected into the fabric. On the other hand, dynamic metrics will cause the priority of a packet to change en route. Leveraging the nonlinear weight (C^h), the different metric can be categorized based on whether C and h are either static or dynamic as summarized in Table I. In describing the different metrics, we assume that a packet is sent from a source node located at (s_x, s_y) to a destination at (d_x, d_y) and the current location is (c_x, c_y). Throughout this work, we assume dimension-ordered routing (DOR) with XY routing.

TABLE I: Arbitration metrics to determine weight of probabilistic arbitration where h is the hop count and C is the contention degree.

h	C	description
static	static	fixed weight (FW)
dynamic	static	constantly increasing weight (CW)
static	dynamic	N/A
dynamic	dynamic	variably increasing weight (VW)

1) Fixed Weight (FW): The total number of hops a packet must travel from its source to its destination is a static value in a mesh network with minimal routing (e.g., dimension-ordered routing). This value is known when the packet is injected into the network. Using this distance, packets which have a longer distance to travel are biased by giving them higher priority at each hop along the way. The static value of the hop count is used based on the source and destination node.

$$h_x = |s_x - d_x|$$
$$h_y = |s_y - d_y|$$

Using these hop count, the weight is calculated according to the dimension being traversed with a contention degree C. While traversing in the x-dimension, $w = 2^{h_x}$ is used and when traversing in the y-dimension, $w = 2^{h_x} \times C^{h_y}$ is used. When traveling in the y-dimension, the weight from the x-dimension is included as well to properly prioritize packets that have traversed longer overall distance. However, the y-dimension weight C^{h_y} is not included while traversing the x-dimension since when a packet only needs to traverse the x-dimension, a packet that needs to traverse both the x and the y-dimension will be unfairly biased. With this metric, the

[2] Simulation setup is described in Section IV.

weight of each packet remains constant or *fixed* throughout the network.

The value of C is dependent on the location of the destination. For a radix-k 2D mesh topology (i.e., $k \times k$ mesh),

$$C = \begin{cases} 3 & d_x = 0 \parallel d_x = k-1 \\ 4 & otherwise. \end{cases}$$

Since 2D mesh is a non-edge symmetric topology, for destination located on the edge of the 2D mesh network, $C = 3$ while $C = 4$ is used for all other destination.

2) Constantly Increasing Weight (CW): Instead of relying on static values, another metric for arbitration is using *dynamic* values, which is based on how much distance a packet has traversed. The distance *traveled* arbitration metric is defined as the number of hops from the current position to the packet source. A packet's weight increases as it gets closer to its destination. The dynamic value of the hop count is from the following.

$$\begin{aligned} h_x &= |c_x - s_x| \\ h_y &= |c_y - s_y| \end{aligned} \quad (1)$$

Similar to the fixed weight (FW) metric, when traveling in the x-dimension, the weight is 2^{h_x} calculated based on the distance traveled (Equation (1)) and when traveling in the y-dimension, $2^{h_x} \times C^{h_y}$ is used where C is based on the destination location as described earlier. When the packet reaches the destination, the weight will be identical to the weight using FW.

Another way to view this metric is to assume that when a packet is injected, it is assigned a weight of 1. As a packet traverses the network, the weight is continually increased. In the x-direction, the weight is multiplied by a factor of 2 at each hop and when traveling in the y-dimension, the weight is multiplied by a factor of C at each hop.

3) Variably Increasing Weight (VW): Instead of a assuming a constant C value at each hop in each dimension, we also evaluate a metric where the value of C per hop is variably changed. Packets are injected with a priority of 1 and the priority also increases dynamically as the packet traverse the network to its destination, similar to CW. However, the increase in weight is not constant as in CW but is dynamic based on the actual contention degree (C) for the output port. The contention degree is defined as the number of packets that are destined for the same router output port. The range of values for C is $1 \leq C \leq (p-1)$ where p is the number of router ports. [3] For example, if there are 3 packets that need to be routed through one output port of a router, each of these packets will have a contention degree of 3. Thus, when these packets are forwarded to the next router, their priorities are increased by 3×. However, if there are no other packets contending for the same output in a given cycle, the weight of the packet remains constant and does not change.

[3]Since we assume no U-turn routing, the maximum value of C is $p-1$.

TABLE II: Synthetic traffic simulation parameters

Parameters	Values
network size	64
topology	2D mesh
routing	XY routing
router latency	3 cycle
buffers	16 flit entry per input port
virtual channels	1
packet size	bimodal(50% 1 flit and 50% 4 flits)

4) Other Metric Considerations: As shown earlier in Table I, another possible weight metric is using static hop count and dynamic contention degree. However, this metric is not applicable since if the hop count is static or determined from the source, the entire weight needs to be fixed at the source – otherwise, the weight would increase by C^h per hop and will not provide EoS as significant more priority is provided to nodes that are farther away. In addition, the dynamic hop count can be obtained from the distance or hop count *remaining*, which is the number of hops to the destination from its current location in the network. This metric would decrease the packet's priority as it approaches the destination. However, decreasing the weight can negate the effect of using probabilistic arbitration. For example, in Figure 4(a), if each packet begins with a fixed weight at its source and if the packet's weights were decreased by $C = 2$ at each hop, the packets that are merged at each router will have equal weights – resulting in an arbitration very similar to round-robin arbitration and not be able to provide any EoS.

IV. Methodology

We evaluate distance-based probabilistic arbitration using a cycle-accurate interconnection network simulator [6]. To evaluate the latency-throughput, the simulator is warmed up under load without taking measurements until steady-state is reached. Then, a sample of injected packets is labeled during a measurement interval. The simulation is run until all labeled packets exit the system. Different synthetic traffic patterns including hotspot traffic, uniform random, bit complement, bit reverse, shuffle, tornado, random permutation, and transpose were used to evaluate probabilistic arbitration. Due to page constraints, only selected results are presented in the next section.

Parameters used in the synthetic simulations are described in Table II. Distance-based, probabilistic arbitration does not require additional VCs so we use a single virtual channel (VC). We assume a FIFO buffer structure – i.e., packet reordering is not allowed at each router input buffer. If additional VCs are required for protocol deadlock, probabilistic arbitration can support additional VCs for different classes of traffic as long as packets stay within the same VCs from source to its destination. The only change required is that VC allocation needs to implement probabilistic arbitration based on distance as well. For the long packets, the head flit goes through switch arbitration using probabilistic arbitration.

The following different arbitration algorithms are compared in the evaluation.

TABLE III: Packet latency variation

	mean(cycles)	max(cycles)	std dev
RR	739	3153	1026
AGE	62.93	63	0.088
VW	62.93	66.2	1.20
CW	62.96	68.8	1.96
FW	62.92	65.5	1.25

- **round-robin arbitration (RR)** : a locally fair, round-robin arbitration at each router node.
- **age-based arbitration (AGE)** : an ideal implementation of age-based arbitration where each packet is timestamped at injection and the age continues to increase every cycle.
- **probabilistic arbitration** : arbitration determined probabilistically by the following different weights.
  - **fixed weight (FW)**
  - **constantly increasing weight (CW)**
  - **variably increasing weight (VW)**

To complement the evaluation of synthetic workload, we also evaluate using traces from PARSEC applications [7] using the Simics/GEMS [8] simulator. We simulate 64 in-order cores with 16 memory controllers, 32KB L1 cache and a shared 16MB L2 cache. We obtain traces from a selected number of benchmarks, including from blackscholes, canneal, and dedup and use the sim-large input dataset.

V. Evaluation

A. Hotspot Traffic

We first evaluate probabilistic arbitration on hotspot traffic where all nodes send traffic to a single destination. As shown in Figure 5, we verify that equality of service is achieved by measuring the accepted throughput across all nodes. The different metrics (FW, CW, VW) all provide very similar results so only the result for CW is shown in Figure 5(d). As a result, by approximating age with hop count and using nonlinear weight with probabilistic arbitration, we can match the performance of age-based arbitration in hotspot traffic and achieve equality of service.

Latency variation is also an important factor in determining overall performance – thus, minimizing the variance is also critical in providing EoS. In Table III, we measure the packet latency variation in hotspot traffic. The packet latency variation is calculated using latency difference for consecutive packets within one flow where a flow is defined as the traffic from a source to the hotspot destination. Age-based arbitration provides the tightest distribution with the lowest variance but all three arbitration weight metric also achieve a very tight distribution with slightly higher variance while the average values are nearly identical. However, locally fair round-robin arbitration not only has a higher mean value but also has a significantly higher variation.

B. Memory Controller Traffic

We also evaluate probabilistic arbitration with multiple hotspot traffic (such as the traffic to memory controller) in future many-core processors. We evaluate a diamond placement of memory controllers [4] with 16 memory controllers and assume a uniform random distribution to 1 of 16 memory controllers. Figure 6 plots the accepted throughput of all the nodes that is sending traffic to the memory controllers. The 16 nodes with zero accepted throughput are the location of the memory controllers. As shown in Figure 6(a), although the diamond placement was shown to provide good performance for on-chip network memory traffic, if round-robin arbitration is used, unfairness is created in reaching the distributed number of MCs – nodes in the middle of the chip are able to send more traffic than the nodes in the corner. Age-based arbitration is able to provide a global fairness and achieve the same throughput for all nodes (Figure 6(b)). Using probabilistic arbitration (Figure 6(c)), we are able to significantly reduce the unfairness compared to round-robin arbitration.

C. Differentiated Service

As shown in the previous subsections, we showed how probabilistic arbitration can provide equality of service. However, some applications will require providing differential services across different nodes – while still providing EoS within a region. By simply modifying the weights of the packet when injected, the priority of the packets can be varied and with probabilistic arbitration, differentiated service can be provided. For example, with the VW metric, packets are injected with a priority of 1. However, if we increase this priority value, then differentiated service can be provided and the amount of differentiated service is proportional to the increase in the initial priority. In Figure 7, we re-simulate the hotspot traffic pattern used in Section V-A but we now partition all nodes into 4 groups – with each group having a priority of 1,2,4,and 8. The results show how each group is able to achieve differentiated service which is proportional to its initial priority while still achieving EoS within each region. Similar results have been shown with different on-chip QoS mechanisms [1] but we show how with probabilistic arbitration and a simple heuristics based on hop count, differentiated service can also be provided without significant complexity.

D. Performance on PARSEC Applications

The performance evaluation using probabilistic arbitration with traces from PARSEC is shown in Figure 8. In Figure 8(a), we compare the performance (network latency) and in Figure 8(b), we compare the network latency in the presence of synthetic hotspot traffic that is added to evaluate the impact of hotspot traffic on the performance of the PARSEC applications. By using probabilistic arbitration (with VW metric), we are able to reduce the interconnect network latency by up to 12% compared to RR (Figure 8(a)). With hotspot traffic, RR arbitration cannot differentiate the traffic but by providing preference to application traffic, we are able to minimize the impact of the hotspot traffic.

E. Performance on Synthetic Traffic Pattern

In this section, we evaluate the impact of probabilistic arbitration on the performance of different synthetic traffic patterns and evaluate its impact on performance. The latency vs. throughput curve for different traffic patterns are shown in Figure 9. Since we only modify the switch arbitration, the

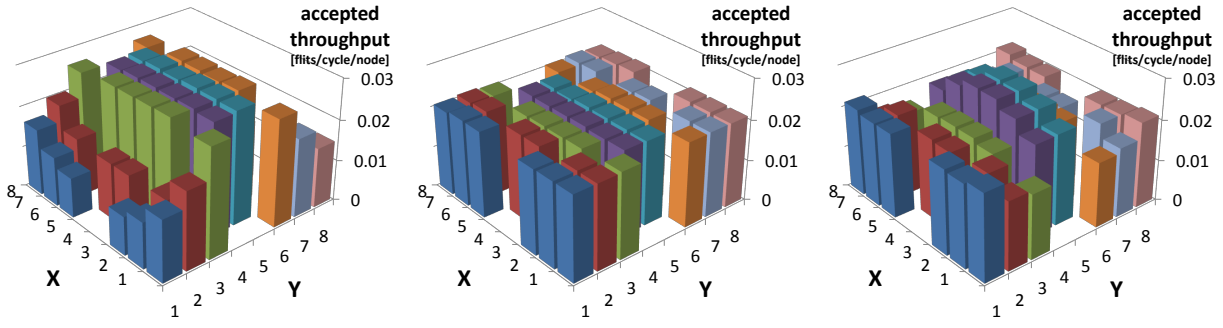


Fig. 6: Performance across multiple hotspots of (a) round-robin, (b) age-based, and (c) probabilistic arbitration.

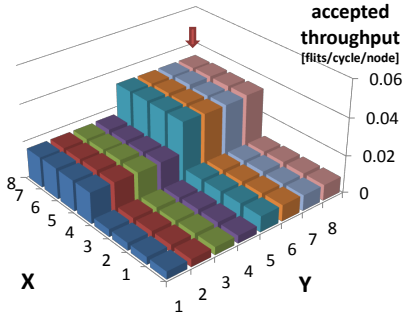


Fig. 7: Hotspot traffic with differentiated service using probabilistic arbitration.

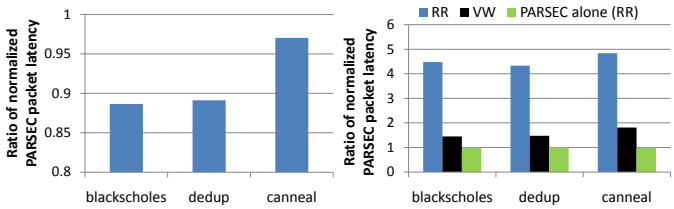


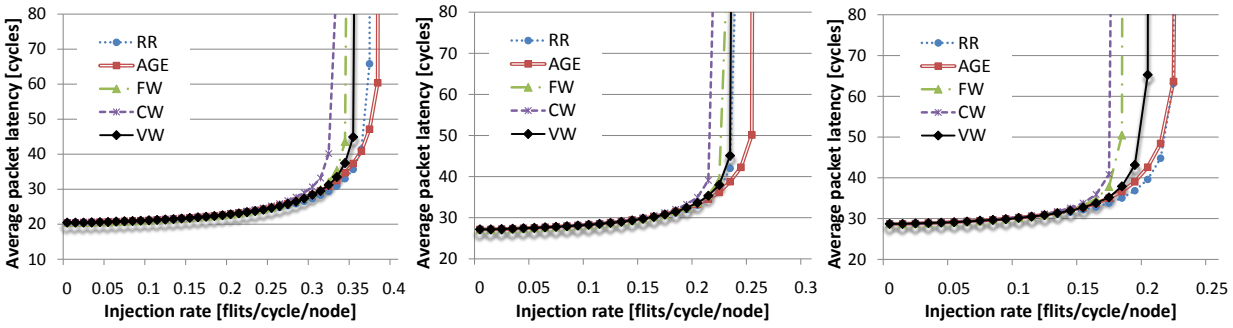
Fig. 8: (a) Performance comparison of PARSEC traces and (b) performance in the presence of hotspot synthetic traffic.

zero-load latencies of the different arbitration are all identical for a given traffic pattern. For some traffic patterns such as bitrev (not shown), all of the different arbitration mechanism achieve nearly identical latency vs. throughput curves. However, for other traffic patterns, the different weight metrics with probabilistic arbitration result in different throughput. For example, with uniform random traffic, CW reduces the saturation throughput by approximately 13% compared to RR while VW and FW provide better performance than CW. For tornado traffic pattern, VW approximately matches the throughput of RR – thus, the ability of providing EoS has minimal impact of performance. Across all traffic patterns, VW generally provides the highest performance compared to FW or CW because of its ability to adapt to the contention by calculating the contention degree at each router before increasing the weight.

In addition to latency vs. throughput curve, we also plot the offered load vs. *minimum* accepted throughput for the different traffic patterns in Figure 10. For traffic patterns such as UR, regardless of the arbitration mechanism, the network continues to accept same amount of traffic past saturation. However, it is known that simple round-robin arbitration can create an *unstable* network for different permutation traffic [6] – i.e., beyond the maximum saturated accepted throughput, as the load continues to increase, the accepted throughput actually *decreases*. By providing globally fairness with age-based arbitration, the maximum accepted throughput can be maintained as offered load continues to increase as shown in Figure 10. For RR, the throughput drop significantly because of global unfairness. The different weight metrics (FW, CW, VW) provide similar saturation throughput but differ significantly on the accepted throughput as load increases beyond saturation. For example, with transpose traffic pattern in Figure 10(a), after saturation around 0.14, as load continues to increase, the throughput drop by approximately 67% for FW while CW and VW maintains stability. For bitcomp (Figure 10(b)), probabilistic arbitration still provides better stability than RR, with VW again providing the highest stability compared to CW and FW. However, VW cannot achieve high sustained throughput as age-based arbitration and it is noticeable in the tornado traffic (Figure 10(c)).

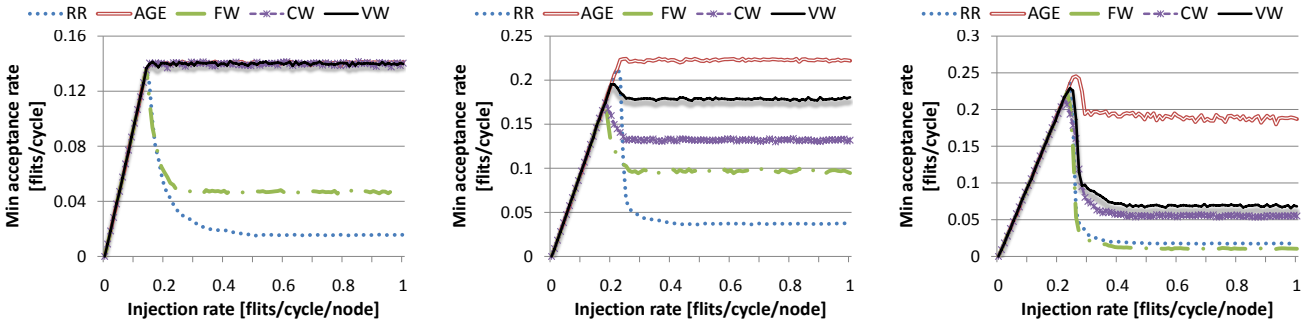
In order to understand the limitations of CW and FW, we use the traffic patterns shown in Figure 11 and Figure 12. Figure 11 highlights the limitation of the FW metric. Assume P1, P2, and P3 sends traffic to P4, P5 and P6, respectively. With this traffic pattern, all of the packets will have a hop count of $h_x = 3, h_y = 0$ and use a weight of $w = 2^3$. As a result, the arbitration at each router (R2, R3, R4, R5) will be round-robin arbitration because of the equal weights. Thus, more bandwidth will be serviced to P3 while the bandwidth used by packets from P1 and P2 will be reduced geometrically – thus, reducing the minimum accepted throughput beyond saturation.

The traffic pattern in Figure 12 highlights the limitation of CW probabilistic arbitration. Assume P0 sends traffic to P7 and P5 sends traffic to P6 and assume the other nodes in the row P1 - P4 are sending traffic to another node in the same column and does not require traversing any channel within this row. With a static constant degree metric using CW, the weight of packet injected at P0 continues to increase and once it reaches R5, it has a weight of 32. However, the



(a) (b) (c)

Fig. 9: Latency throughput curve for (a) uniform random, (b) tornado, and (c) bitcomp traffic patterns.



(a) (b) (c)

Fig. 10: Offered load vs. minimum accepted throughput for (a) transpose, (b) bitcomp, and (c) tornado traffic patterns.

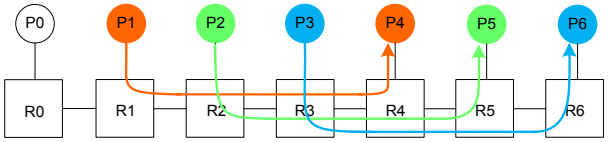


Fig. 11: Traffic pattern that highlights the limitation of the fixed weight probabilistic arbitration.

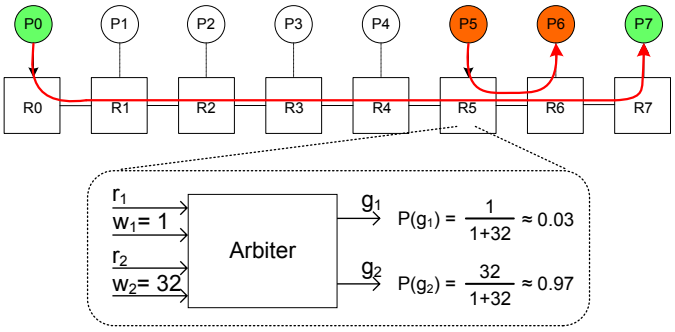


Fig. 12: Traffic pattern that highlights the limitation of the constantly increasing weight probabilistic arbitration. r_1, w_1 represents a packet from P0 and r_2, w_2 represents a packet from P5.

packet injected from P5 at R5 will have weight of 1. Thus, using probabilistic arbitration, P0 will receive 32/33 of the bandwidth from the channel between R5 and R6 while P5 will only obtain 1/33 of the bandwidth – unfairly, biasing the packet that have traveled long distance. Ideally, since there is only 2 flow sharing the channel between R5 and R6, each should access 1/2 the bandwidth. In order to overcome the limitation of CW, *variably* increasing weight metric is needed. Thus, for packet that is injected at P0, it does not encounter any contention until it reaches R5 and maintains a weight of 1. At R5, $w1 = w2 = 1$ and each flow from P0 and P5 will be serviced approximately equally.

F. Performance Comparison to GSF

In this section, we compare the performance of probabilistic arbitration described in this work with a QoS scheme for on-chip networks, GSF(globally synchronized frames) [1]. GSF takes a *frame-based* approach as time is coarsely quantized into frames, and injection control logic at each source node controls bandwidth allocation by restricting the number of flits that each traffic flow can inject into each frame.

Figure 13 shows minimum accepted throughput versus injection rate using both GSF and probabilistic arbitration (VW) with multiple buffer configurations for three traffic patterns. The throughput of GSF suffers when the buffer size is small and clearly shows how GSF is sensitive to the number of virtual channels (VCs). For example, when the number of VCs is only two (GSF(2×4) in Figure 13), the network utilization of GSF is low because there is only one future frame (assuming one VC per frame), which is not sufficient to hide the drain time of the head frame. On the other hand, probabilistic arbitration shows generally higher minimum throughput for the same buffer size (e.g., GSF(2×4) vs. VW(1×8) and GSF(4×4) vs. VW(1×16)). However, for traffic such as tornado traffic, the performance of VW and probabilistic arbitration suffers

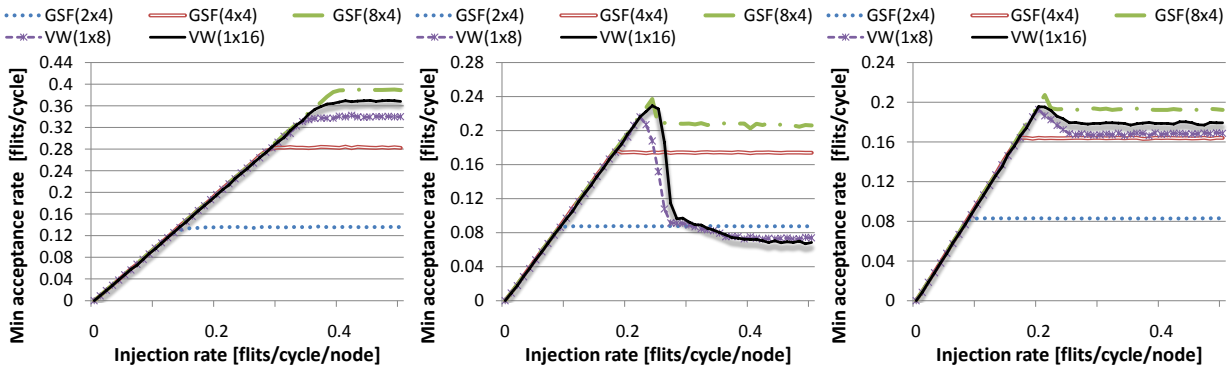


Fig. 13: Performance comparison using GSF and probabilistic arbitration for (a) uniform random, (b) tornado, and (c) bitcomp traffic pattern. $V \times B$ in parentheses indicates buffer configuration: the number of virtual channels (VCs) per physical link (V) $\times$ buffer size per VC (B).

at high load because of the limitation of *contention*-based arbitration that is used with VW weight metric.

VI. Discussion

A. Starvation and Livelock

As mentioned earlier in Section III-A, probabilistic arbiter can create starvation. For example, in Figure 2(a), the probability that r_1 with w_1 will not be serviced for n consecutive cycles is $(1 - P(g_1))^n$, assuming r_2 also has a request for n consecutive cycles with the same weight w_2. Theoretically, as $n \rightarrow \infty$, all requests will be eventually served but n can get very large. In Figure 14, we plot the probability of a packet not being serviced for n consecutive cycles as we vary $P(g_i)$. For $P(g_i) > 0.1$, the probability quickly converges to zero and there is minimal impact of starvation using probabilistic arbitration. For $P(g_i) \leq 0.1$, in the worst case, there is chance that a packet will not be serviced for large number of cycles and will get only worse with smaller values of $P(g_i)$.

However, even with a globally fair arbitration such as age-based arbitration or weighted fair queueing [9], in a traffic pattern like hotspot traffic, each node's traffic will only be serviced every N cycles where N is the number of nodes in the network. Thus, for $N = 64$, the probability of a packet being serviced within 64 cycles ($n = 64$) with probabilistic arbitration ($P(g_i) = 0.01$) will be approximately 0.5. In addition, Figure 14 is an *upper bound* on the probability of a packet not being serviced as we assume other requests are continually asserted such that $P(g_i)$ remains constant. If in the next cycle the number of requests is reduced, $P(g_i)$ will increase – thus, reducing the probability of a packet not being serviced.

In Figure 15, we plot the total number of times (or cycles) each packet loses arbitration for the different weight metrics on the x-axis and the percentage of packets in the y-dimension. We obtain this metric by counting the total number of requests made to the probabilistic arbiter for each packet en route to its destination and subtract the hop count from source to its destination, which corresponds to the number of successful arbitration. We use the hotspot traffic from Section V-A since

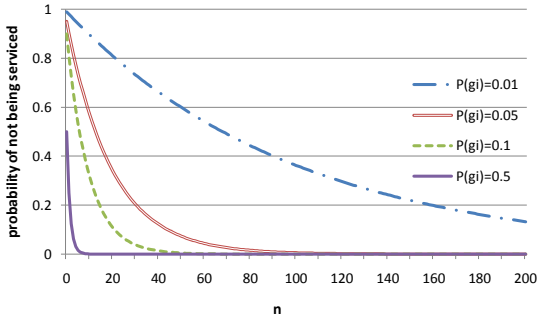


Fig. 14: Probability of packet not being serviced for n consecutive cycles.

that represents a worst-case traffic pattern for starvation as some packet needs to traverse the maximum network diameter to reach its destination – resulting in the highest weight and creating the highest probability of starvation for packets near the hotspot destination. We use an injection rate of 0.015 at each node which approximately corresponds to the maximum achievable throughput for each node with the hotspot traffic ($\approx 1/64$). As shown in Figure 15, FW and CW take very long for it converge to 100% while with VW, the convergence occurs much faster as the number of times a packet loses probabilistic arbitration is under 20. With VW, packet's weight does not always constantly increase and minimizes packets with large values – resulting in higher values of $P(g_i)$ and fewer number of cycles waiting for lost arbitration.

However, if starvation avoidance needs to be guaranteed, the probabilistic arbiter can have a fall-back mechanism – for example, if a request is not serviced for n cycles, the arbiter falls back to a simple round-robin arbiter for some number of cycles to ensure that everyone does get serviced at least once every n cycles.

B. Implementation

To support probabilistic arbitration, the only change required in an on-chip network router is the switch arbitration. The cost (in terms of area and power) of an on-chip router is dominated by the buffers and the crossbar [10], [11], [12]. Prior work have shown that the area and power consumption

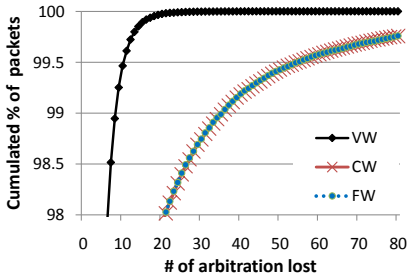


Fig. 15: Number of cycles packets wait before being served with probabilistic arbitration for hotspot traffic with the different weight metrics.

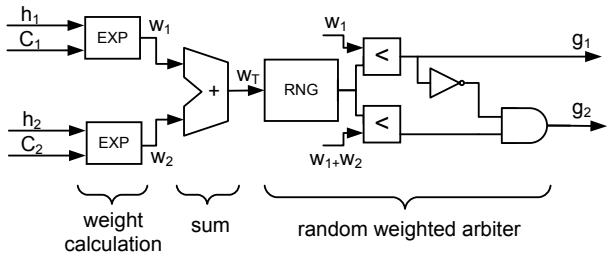


Fig. 16: Block diagram of a probabilistic arbiter implementation.

impact of arbitration is minimal [13] – for example, the power consumption of arbitration was approximately 2.5% of the total power consumption [14]. Thus, the additional complexity of implementing distance-based, probabilistic arbitration on the router area and energy should be relatively small. However, the latency of arbitration is often the critical path in a router [6] and if not implemented properly, probabilistic arbiter can create a bottleneck.

A block diagram of a probabilistic arbiter is shown in Figure 16. [4] The arbiter is broken into three steps : 1) weight calculation, 2) adding all the weights together, and 3) the random weighted arbiter. In the first step, the nonlinear weights need to be calculated based on the h and C parameters and an exponential calculation (EXP) is needed. In the second step, all the weights are summed together to calculate w_T. Based on this value w_T, the last step involves generate a random number (RNG) between 0 and $w_T - 1$ and depending on the range of this random number, the appropriate grant is asserted.

In this section, we focus on VW instead of CW or FW because of its performance advantage. The critical path shown in Figure 16 can be reduced using different techniques. For a random number generator (RNG), a linear feedback shift register can be implemented. However, the random number is dependent on the w_T which increases the critical path. To avoid this critical path, the random number can be pregenerated based on the maximum weight possible in the network. Once w_T is calculated, the random value can be selected based on using the lower $log(w_T)$ bits of the pregenerated random number. By using this technique, we estimate the arbitration delay of probabilistic arbitration (VW) to be approximately

[4]For simplicity, an arbiter with only two requests is shown. In a 2D mesh topology, the arbiter will need to support up to 4 requests since U-turns are not allowed.

	age-based arbitration	VW
arbitration	deterministic	probabilistic
weight metric	age, clock	hop count, contention
weight management	global management needed within each router to track packets in each epoch	per-packet based
weight rollover	counter saturation, starvation	not needed
weight update	every clock cycle	every hop

TABLE IV: Qualitative comparison of age-based arbitration and VW algorithm. Age-based arbitration is based on the implementation from Cray XT3 [5].

15 FO4 which includes the adder, mux, comparator, and the selector logic, compared with a conventional arbiter at 10 or 11 FO4 [15]. The critical path can be further reduced by trading-off complexity and accuracy. For example, the weights can be approximated by using only m of n bits. Additional design trade-off can be made between router complexity and on-chip bandwidth – i.e., increase router complexity by completely recalculating the weight at each router or increase on-chip bandwidth usage by carrying around the weight (or partial weight) within the head flit as this simplifies the calculation of the new weight.

In addition, we qualitatively compare VW with age-based arbitration in Table IV. Assuming both arbitration carry a n-bit field to represent the weight, the main difference is how this weight maintained. For VW, this weight is only updated once per router and does not need any special maintenance. However, with age-based arbitration, the age field will eventually saturate and reach the maximum value. As a result, careful maintenance is need such that when the counter does roll-over, proper age is maintained while starvation is avoided [5].

VII. Related Work

Probabilistic techniques for centralized arbitration/scheduling have been proposed in OS scheduling and system-on-chip shared bus system. Lottery scheduling [16] chooses a thread to run using random numbers and Lotterybus [17] uses probabilistic arbitrations to choose the owner of a shared bus. Probabilistic arbitration has also been proposed within memory schedulers [18]. These works have a single centralized arbiter/scheduler, however, our work uses multiple distributed probabilistic arbiters in on-chip networks and we presents novel weight metrics to achieve fairness with probabilistic arbiter. Distance or hop count was also used in the arbitration within Aérgia architecture [19] where they used hop count to determine *slack* calculation and provide application-level fairness.

Although EoS has been well investigated in other fields such as computer networking and real-time system, their solutions cannot be easily applied to on-chip environment because of different constraints, compared to off-chip. We divide the solution space into two classes. The first class of approaches is based on injection rate control. Injection rate control can be placed at either the injection point of each source or the input channel of each intermediate node to limit the maximum number of flits a network or an individual node can service for

each flow over a period of time. This time period of bandwidth accounting is called *frame* in some proposals [20], [1], [3]. Æthereal [21] uses pipelined time-division-multiplexed (TDM) circuit switching to implement guaranteed performance services. Each flow is required to explicitly set up a channel on the routing path before transmitting the first payload packet, and a flow cannot use more than its fair bandwidth share even if the network is underutilized. SonicsMX [22] can support EoS without explicit channel setup. However, each node has to maintain per-thread queues, which make it only suitable for a small number of threads. QNoC [23] takes a source regulation approach and requires each source to fetch credit tokens from the destination (hotspot) node before sending out payload packets. It requires only minimal modifications to network routers because most of the intelligence is at end nodes. However, QNoC requires a sophisticated secondary network (either logical or physical) for credit token request/reply not to slow down the source injection process and potentially penalizes short-lived flows because of credit token fetch.

The second class of approaches proposes sophisticated arbitration techniques to provide EoS. (Weighted) Fair Queueing [9] and Virtual Clock [24] are best-known queueing and scheduling algorithms in this class. They are developed for EoS in long-haul IP networks where large buffers are available. These achieve fairness and high network utilization, but each router is required to maintain per-flow state and queues which would be impractical in an on-chip network. The MediaWorm router [25] evaluates the performance of both Fair Queueing and Virtual Clock in a multiprocessor environment using multimedia traffic. Weighted round-robin arbiter [6] is a degenerated form of Fair Queueing, which does not take packet size into account.

VIII. Conclusions

In this work, we presented distance-based, probabilistic arbitration to provide equality-of-service (EoS) in many-core CMPs. By only modifying the arbitration of on-chip network routers, we showed how the proposed arbitration can approach the behavior of ideal age-based arbitration without requiring any significant complexity and relying only on local arbitration. We described how simple heuristics based on hop count can be used to approximate age. Along with a probabilistic arbiter, we were able to achieve livelock-free arbitration that avoids starvation. Three different arbitration weight metrics including fixed weight, constantly increasing weight, and variably increasing weight were described which rely on nonlinear weights to provide EoS to nodes that are farther away. Our simulation results also show how there is minimum degradation of performance across a wide range of traffic patterns in evaluating non-EoS aspect of using probabilistic arbitration. By providing better global fairness, distance-based, probabilistic arbitration is also able to provide a more stable network as it is able to continue delivering throughput that is close to the peak throughput when the offered load is beyond the saturation throughput.

References

[1] J. W. Lee, M. C. Ng, and K. Asanovic, "Globally-synchronized frames for guaranteed quality-of-service in on-chip networks," in *ISCA*, Beijing, China, 2008, pp. 89–100.

[2] B. Grot, S. W. Keckler, and O. Mutlu, "Preemptive virtual clock: A flexible, efficient, and cost-effective qos scheme for networks-on-a-chip," in *MICRO*, New York, NY, 2009.

[3] D. Stiliadis and A. Varma, "Design and analysis of frame-based fair queueing: a new traffic scheduling algorithm for packet-switched networks," in *SIGMETRICS*, Philadelphia, PA, 1996, pp. 104–115.

[4] D. Abts, N. D. Enright Jerger, J. Kim, D. Gibson, and M. H. Lipasti, "Achieving predictable performance through better memory controller placement in many-core cmps," in *ISCA*, Austin, TX, 2009, pp. 451–461.

[5] D. Abts and D. Weisser, "Age-based packet arbitration in large-radix k-ary n-cubes," in *SC '07*, Reno, Nevada, 2007, pp. 1–11.

[6] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. San Francisco, CA: Morgan Kaufmann, 2004.

[7] C. Bienia, S. Kumar, J. P. Singh, and K. Li., "The PARSEC Benchmark Suite: Characterization and Architectural Implications," in *International Conference on Parallel Architectures and Compilation Techniques*, 2008.

[8] M. M. K. 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," *SIGARCH Comp. Arch. News*, vol. 33, no. 4, pp. 92–99, 2005.

[9] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *SIGCOMM*, Austin, Texas, 1989, pp. 1–12.

[10] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, and N. Borkar, "An 80-tile 1.28TFLOPS network-on-chip in 65nm CMOS," *ISSCC*, pp. 98–589, Feb. 2007.

[11] P. Gratz, C. Kim, R. McDonald, S. Keckler, and D. Burger, "Implementation and Evaluation of On-Chip Network Architectures," in *ICCD*, 2006.

[12] D. Wentzlaff, P. Griffin, H. Hoffmann, L. Bao, B. Edwards, C. Ramey, M. Mattina, C.-C. Miao, J. Brown, and A. Agarwal, "On-chip interconnection architecture of the tile processor," *IEEE Micro*, vol. 27, no. 5, pp. 15–31, Sept.-Oct. 2007.

[13] H. Wang, L. S. Peh, and S. Malik, "Power-driven Design of Router Microarchitectures in On-chip Networks," in *MICRO*, 2003, pp. 105–116.

[14] A. Kumar, P. Kundu, A. Singh, L.-S. Peh, and N. Jha, "A 4.6tbits/s 3.6ghz single-cycle noc router with a novel switch allocator in 65nm cmos," in *ICCD*, October 2007.

[15] L.-S. Peh and W. J. Dally, "A delay model and speculative architecture for pipelined routers," in *HPCA*, 2001.

[16] C. A. Waldspurger and W. E. Weihl, "Lottery scheduling: flexible proportional-share resource management," in *OSDI*, 1994.

[17] K. Lahiri, A. Raghunathan, and G. Lakshminarayana, "Lotterybus: a new high-performance communication architecture for system-on-chip designs," in *DAC*, Las Vegas, Nevada, 2001, pp. 15–20.

[18] I. Hur and C. Lin, "Adaptive history-based memory schedulers," in *MICRO*, 2004, pp. 343–354.

[19] R. Das, O. Mutlu, T. Moscibroda, and C. R. Das, "Aérgia: exploiting packet latency slack in on-chip networks," in *ISCA*, Saint-Malo, France, 2010, pp. 106–116.

[20] J. H. Kim and A. A. Chien, "Rotating Combined Queueing (RCQ): Bandwidth and latency guarantees in low-cost, high-performance networks," in *ISCA*, Philadelphia, PA, 1996, pp. 226–236.

[21] K. Goossens, J. Dielissen, and A. Radulescu, "Æthereal network on chip: Concepts, architectures, and implementations," *IEEE Des. Test*, vol. 22, no. 5, pp. 414–421, 2005.

[22] W.-D. Weber, J. Chou, I. Swarbrick, and D. Wingard, "A quality-of-service mechanism for interconnection networks in system-on-chips," in *DATE*, Munich, Germany, 2005, pp. 1232–1237.

[23] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "QNoC: QoS architecture and design process for network on chip," *J. Syst. Archit.*, vol. 50, no. 2-3, pp. 105–128, 2004.

[24] L. Zhang, "Virtual Clock: a new traffic control algorithm for packet switching networks," in *SIGCOMM*, Philadelphia, PA, 1990, pp. 19–29.

[25] K. H. Yum, E. J. Kim, C. R. Das, and A. S. Vaidya, "MediaWorm: A QoS capable router architecture for clusters," *IEEE Trans. Parallel Distrib. Syst.*, vol. 13, no. 12, pp. 1261–1274, 2002.