Novel Flow Control for Fully Adaptive Routing in Cache-coherent NoCs

Sheng Ma, Zhiying Wang, *Member, IEEE,* Natalie Enright Jerger, *Senior Member, IEEE,*
Li Shen and Nong Xiao, *Member, IEEE*

Supplementary file

Abstract—This paper contains the supplementary material to the paper submitted to IEEE TPDS entitled "Novel Flow Control for Fully Adaptive Routing in Cache-coherent NoCs" (labeled as "main file" in this paper). In Sec. 1, we prove that if a routing algorithm with WPF is deadlock-free, then this algorithm with conservative VC re-allocation is also deadlock-free. In Sec. 2, we provide further insights for the performance trends of routing algorithms by analyzing the buffer utilization. In Sec. 3, we present the full system performance analysis. In Sec. 4, we show the recursive algorithm to calculate the allowable escape VC. In Sec. 5, we perform sensitivity study on several configuration parameters. In Sec. 6, we provide further discussion about the designs.

1 Logical Equivalence of Alg and Alg+WPF

Sec. 4.1 proposes Theorem 1, which declares that if a fully adaptive routing algorithm with conservative VC re-allocation (named as Alg) is deadlock-free, then applying WPF on this routing algorithm (named as Alg+WPF) is also deadlock-free. This section proves that if Alg+WPF is deadlock-free, then Alg is deadlock-free as well. In addition to Assumption 1 in Sec. 4.1 of the main file, this proof needs another simple assumption about the routing algorithm.

***Assumption* 2.** The definition of a routing algorithm is independent of the packet length.

Assumption 2 implies that if two packets are injected from the same source and destined to the same node, they can use the same set of VCs even though these packets have different lengths. This assumption is generally used in many previous theories [2], [3], [6], [7], [8], [9], [10], [12], [18], [21], [24], [25]. We prove the following theorem.

***Theorem* 3**: If Alg+WPF is deadlock-free, then Alg is also deadlock-free.

Description: This proof is similar to the proof of Theorem 1 in the main file. We prove that if Alg has a deadlock configuration, then Alg+WPF has a deadlock configuration as well. Use $Config_0$ in Fig. 1 as an example. We replace packet P_i with packet P_i' to completely fill all VCs. The source and destination of P_i' are the same as P_i, while P_i' may have more flits than P_i to completely fill the VCs. We prove

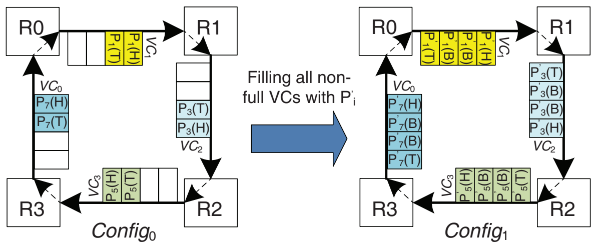


Fig. 1. The construction of a new configuration.

that Alg+WPF can achieve $Config_0$, and $Config_0$ is a deadlock configuration. However, Alg+WPF is deadlock-free, thus there is no such configuration.

Proof: By contradiction. If Alg is not deadlock-free, then there is a deadlock configuration ($Config_0$) in which a set of packets, P_{set_0}, are waiting on VCs held by other packets in P_{set_0}. We prove that there is a deadlock configuration for Alg+WPF in three steps.

Step 1: We build a new configuration based on $Config_0$. Consider each packet P_i in P_{set_0}. If P_i does not completely fill all VCs where its flits reside, we replace P_i with packet P_i' to completely fill all VCs occupied by P_i. The source and destination of P_i' are the same as P_i, and the length of P_i' is larger than P_i. If P_i completely fills all VCs where its flits reside, P_i' is the same as P_i. We label the new configuration as $Config_1$, and the set of packets in $Config_1$ as P_{set_1}.

Step 2: We prove that when the network is routed by Alg+WPF, all packets in P_{set_1} can move to their current VCs in $Config_1$, and the head flits of these packets are at VC heads. For each packet P_i' in P_{set_1}, we consider its corresponding packet P_i in $Config_0$. We further consider each hop hop_k of P_i when it is routed by Alg. Assume the head flit of P_i moves to VC_k's head during hop_k. The routing algorithm allows P_i to use VC_k. Although P_i' and P_i may have different packet lengths, they have the same source and destination information. Based on Assumption 2, the routing algorithm allows P_i' to use VC_k as well.

Since Alg+WPF is deadlock-free, VC_k must become available at some time. Based on Assumption

• Sheng Ma, Zhiying Wang, Li Shen and Nong Xiao are with the State Key Laboratory of High Performance Computing, National University of Defense Technology, China. Sheng Ma is also with the National Key Laboratory for Paralleling and Distributed Processing, NUDT.
E-mail: {masheng, zywang, lishen, nongxiao}@nudt.edu.cn
This research was mostly done when Sheng Ma was a visiting international student at the University of Toronto.
• Natalie Enright Jerger is with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ontario, Canada.
E-mail: enright@eecg.toronto.edu

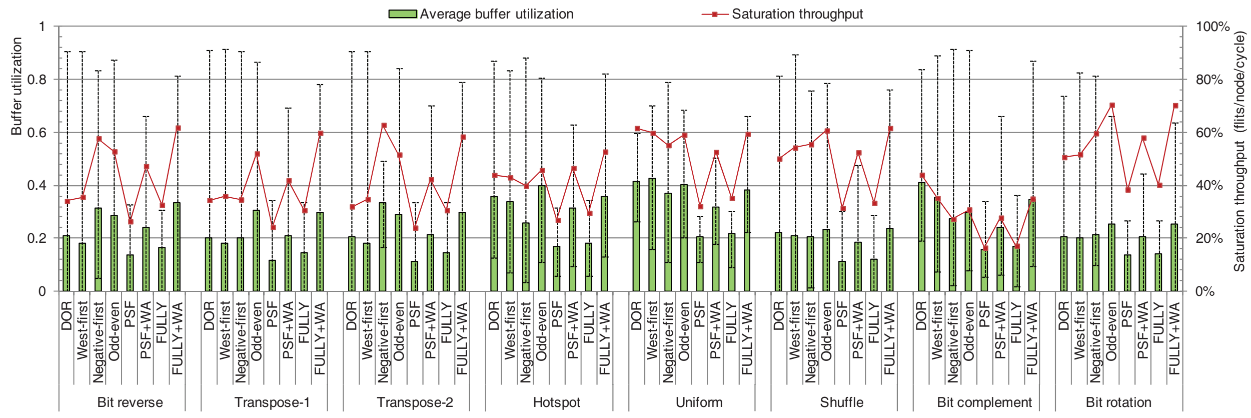


Fig. 2. The buffer utilization and throughput in saturation for several routing algorithms.

1 in the main file, P_i' has some possibility to request VC_k. There are two cases when P_i' requests VC_k.

2.1) VC_k is empty. VC_k can be re-allocated to P_i' with conservative VC re-allocation. Thus, the head flit of P_i' will move to VC_k's head.

2.2) VC_k is not empty. But VC_k already received the tail flit of last allocated packet and still has enough buffers for P_i'. WPF can re-allocate VC_k to P_i'. Yet, when P_i' reaches VC_k, its head flit is not at VC_k's head. Considering Alg+WPF is deadlock-free, the packets in VC_k before P_i' must be sent out in limited time. Then, the head flit of P_i' is at VC_k's head.

Considering 2.1) and 2.2) together, if the head flit of P_i moves to the head of a VC during any hop when routed by Alg, the head flit of P_i' can also move to the same VC head when routed by Alg+WPF. Thus, when routed by Alg+WPF, P_i' can be routed to its current VC(s) in $Config_1$, and the head flit of P_i' is at the VC head.

Step 3: We prove that $Config_1$ is a deadlock configuration for Alg+WPF. Since all VCs in $Config_1$ are completely filled, all packets in P_{set_1} still wait for VCs held by other packets in P_{set_1}, even with WPF applied. $Config_1$ is a deadlock configuration for Alg+WPF. But $Alg + WPF$ is deadlock-free, so there is no deadlock configuration. Thus, Alg is deadlock-free as well. □

Theorem 1 of the main file declares that if Alg is deadlock-free, then Alg+WPF is deadlock-free as well. Theorem 3 declares that if Alg+WPF is deadlock-free, then Alg is deadlock-free as well. Considering these two theorems together, the deadlock freedom of Alg+WPF is logically equivalent to the deadlock freedom of Alg. Therefore, the WPF optimization of traditional wormhole flow control does not have any influence on the deadlock freedom.

2 Buffer Utilization of Routing Algorithms

Sec. 5.1 of the main file measures the performance of several fully adaptive, partially adaptive and deterministic routing algorithms. This section provides further insights of the performance trends by analyzing the buffer utilization of all evaluated designs. Fig. 2 illustrates the average buffer utilization of all network VCs in saturation for eight synthetic traffic patterns. The maximum and minimum rates are given by the error bars. Fig. 2 also shows the saturation throughput supported by routing algorithms. There are several important insights.

First, buffers are used to support high throughput. The saturation throughput of routing algorithms is related to the buffer utilization. Higher average buffer utilization generally leads to higher saturation throughput. This trend is obvious in bit complement. Bit complement sends traffic from node $\{s_3,s_2,s_1,s_0\}$ to node $\{\neg s_3,\neg s_2,\neg s_1,\neg s_0\}$, which has the largest average hop count [14], and puts the highest pressure on the buffers among all evaluated traffic patterns.

Second, for the same routing algorithm in different traffic patterns, there may be fluctuations between the buffer utilization and the saturation throughput. For example, even though DOR shows much higher buffer utilization for bit complement than shuffle, its saturation throughput for bit complement is lower than shuffle. These fluctuations are due to that the same routing algorithm uses different VCs for different patterns; the network becomes saturated in different status. Similarly, since some algorithms use different VCs in the same traffic pattern, there may be slight fluctuations as well. For example, even though the average buffer utilization of west-first is slightly lower than DOR for bit reverse, its saturation throughput is higher than DOR.

Third, since all fully adaptive routing algorithms, including PSF, PSF+WA, FULLY and FULLY+WA, use the same set of VCs for the same traffic pattern, their saturation throughput is roughly proportional to the buffer utilization. For instance, in bit reverse, PSF+WA's average buffer utilization is 78.3% higher than PSF, making it perform 79.5% better than PSF. Similarly, PSF+WA has 67.4% higher average buffer utilization than PSF in shuffle, and its saturation throughput is 67.2% higher than PSF.

Fourth, the network becomes saturated when some resources become saturated [4]. For most patterns and

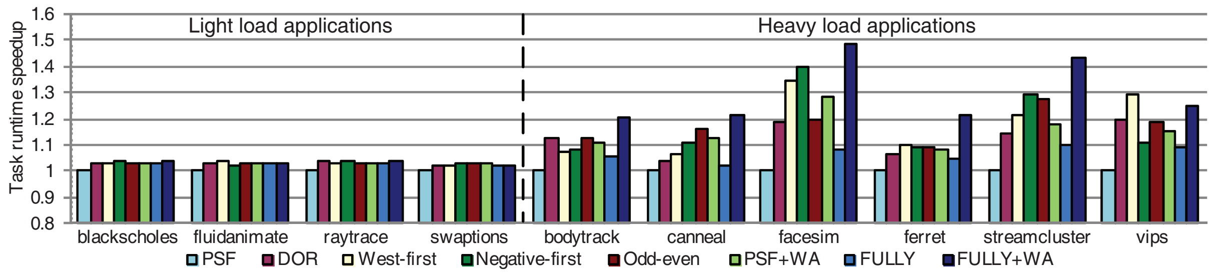


Fig. 3. System speedup for PARSEC benchmarks.

most algorithms, the bottleneck is VCs; networks become saturated when some VCs have more than 80% utilization, including DOR, west-first, negative-first and odd-even in most patterns. When the algorithm provides abundant adaptiveness, it distributes traffic more uniformly among VCs, preventing them from becoming the bottleneck. Instead, the bottleneck may be the crossbar or network interface [23]. For example, since negative-first offers adaptiveness for all traffic in tranpose-2, the network becomes saturated even when the maximum buffer utilization is only 49.2%. Similarly, when in saturation, the maximum buffer utilization of PSF+WA and FULLY+WA is lower than partially adaptive routing and DOR.

Fifth, PSF and FULLY are limited by conservative VC re-allocation; their maximum buffer utilization for evaluated patterns is less than 40%. Applying WPF on adaptive VCs and aggressive VC re-allocation on escape VCs greatly increases buffer utilization, which proportionally improves performance. FULLY supports higher routing flexibility than PSF, which brings higher buffer utilization and performance.

3 Evaluation on PARSEC Workloads

3.1 Methodology and Configuration

To measure full-system performance, we leverage FeS2 [16] for x86 simulation and BookSim for NoC simulation. FeS2 is implemented as a module for Simics [15]. We run PARSEC benchmarks [1] with 16 threads on a 16-core CMP, which is organized as a 4×4 mesh. Prior research shows the frequency of simple cores in a many-core platform can be optimized to 5∼10 GHZ, while the frequency of NoC router is limited by the allocator speed [5]. We assume cores are clocked 5× faster than the network. Cache lines are 64 bytes; long packets are 5 flits long with a 16-byte flit width. We use a distributed, directory-based MOESI coherence protocol which needs 4 VNs for protocol-level deadlock freedom. Each VN has 2 VCs; each VC is 4 flits deep. The router pipeline is the same as described in Sec. 4.4 of the main file. All benchmarks use the *simsmall* input sets. The total runtime is the performance metric. TABLE 1 gives the system configuration.

3.2 Performance

Fig. 3 shows the speedups relative to PSF for PARSEC workloads. We divide the 10 applications into 2 classes. For `blackscholes`, `fluidanimate`, `raytrace` and `swaptions`, different algorithms perform similarly. The working sets of these applications fit into the caches and their computation phases consist of few synchronization points, leading to a lightly loaded network. The system performance of these applications is unaffected by techniques that improve the network throughput, such as sophisticated routing algorithms.

TABLE 1. Full system simulation configuration.

# of cores	16, 4×4 Mesh
L1 cache (D & I)	private, 4-way, 32KB each
L2 cache	private, 8-way, 512KB each
Cache coherence	MOESI distributed directory

However, the routing algorithms affect the other 6 applications. These applications have heavier loads than previous 4 applications. Yet, their average aggregate loads during the running periods are lower than the network saturation points. The highest average aggregate injection rate for `canneal` is 12.3% flits/node/cycle, which is below the saturation points for all designs. Two factors bring performance improvements for these 6 applications. First, these applications exhibit period bursty communication and the synchronization primitives create one hotspot inside the network. The bursty communication and the hotspot make the network operate past saturation at times during the running period. Thus, these applications benefit from routing algorithms with higher throughputs. Second, short packets are critical in cache-coherent many-core platforms; they carry time-critical control messages that are often on the application's critical path. These short packets can affect the execution time significantly. The novel flow control reduces the packet latency, especially under heavy loads, which brings performance gains. For example, FULLY+WA has 48.5% and 43.0% speedups over PSF for `facesim` and `streamcluster`.

Since most of `vips` application's bursty communication is eastbound, west-first performs best for `vips`. For `facesim` and `streamcluster`, negative-first offers higher adaptiveness than odd-even, thus achieving better performance. For all heavy load applications except `vips`, FULLY+WA performs best. Across these applications, FULLY+WA achieves an average of 21.3% and maximum 37.8% speedup over FULLY. With sufficient flexibility, FULLY+WA has an average 12.1% speedup over PSF+WA. The average speedups of FULLY+WA are 29.3%, 15.0%, 10.1%,

```
void AllowableEVCs(int cx,int cy,int dx,int dy,
                   int*** EVCs){
   if( cx == dx && cy == dy ){  // destination
     return;
   } else if( cx < dx && cy == dy ){   // east
     EVCs[cx+1][cy][1] = 1;
     AllowableEVCs( cx+1, cy, dx, dy, EVCs );
   } else if( cx > dx && cy == dy ){   // west
     EVCs[cx-1][cy][0] = 1;
     AllowableEVCs( cx-1, cy, dx, dy, EVCs );
   } else if( cx == dx && cy < dy ){   // north
     EVCs[cx][cy+1][3] = 1;
     AllowableEVCs( cx, cy+1, dx, dy, EVCs );
   } else if( cx == dx && cy > dy ){   // south
     EVCs[cx][cy-1][2] = 1;
     AllowableEVCs( cx, cy-1, dx, dy, EVCs );
   } else if( cx>dx && cy>dy ){   // north-west
     EVCs[cx-1][cy][0] = 1;
     AllowableEVCs( cx-1, cy, dx, dy, EVCs );
     AllowableEVCs( cx, cy-1, dx, dy, EVCs );
   } else if( cx>dx && cy<dy ){   // south-west
     EVCs[cx-1][cy][0] = 1;
     AllowableEVCs( cx-1, cy, dx, dy, EVCs );
     AllowableEVCs( cx, cy+1, dx, dy, EVCs );
   } else if( cx<dx && cy>dy ){   // north-east
     EVCs[cx+1][cy][1] = 1;
     AllowableEVCs( cx+1, cy, dx, dy, EVCs );
     AllowableEVCs( cx, cy-1, dx, dy, EVCs );
   } else if( cx<dx && cy<dy ){   // south-east
     EVCs[cx+1][cy][1] = 1;
     AllowableEVCs( cx+1, cy, dx, dy, EVCs );
     AllowableEVCs( cx, cy+1, dx, dy, EVCs );
   }
}
```

Fig. 4. The `AllowableEVCs` algorithm. The initial value of `EVCs` is 0s. `cx` and `cy` are X and Y positions of the current node. `dx` and `dy` are X and Y positions of the destination. Port encoding: East: 0, West: 1, South: 2, North: 3.

9.9% and 10.4% over PSF, DOR, west-first, negative-first and odd-even, respectively.

4 Allowable Escape VCs (EVCs)

In Sec. 5.2 of the main file, we leverage a recursive algorithm to calculate the allowable escape VCs (EVCs) for synthetic traffic patterns. Here, we present the algorithm, as shown in Fig. 4.

This algorithm recursively calculates all allowable EVCs for a packet sent from node (cx, cy) to node (dx, dy). At each step, the algorithm marks the escape VC at the port which adheres to DOR as allowable by setting the corresponding element of $EVCs$ array to '1'. Fully adaptive routing may forward the packet to all neighbors inside the minimum quadrant defined by the current position and the destination. At the next step, the algorithm continues by using these neighbors as current positions. For example, if $cx>dx$ and $cy>dy$, then the destination is at the north-west quadrant of current position. The EVC at the east input port of node $(cx-1, cy)$ is allowed to be used. The packet may move to node $(cx-1, cy)$ and node $(cx, cy-1)$, and the algorithm continues by using them as current positions at the next step. By considering all source and destination pairs defined in the traffic pattern, we get the allowable EVCs in a 4×4 mesh network, as shown in TABLE 4 of the main file.

5 Sensitivity to Network Design

This section performs sensitivity study for network configuration parameters. Except for the analyzed parameter, other parameters are the same as the baseline configuration shown in TABLE 2. This section makes comparison with the performance of baseline configuration shown in the main file. To improve the readability, we reproduce Fig. 10 of the main file, as shown in Fig. 5 here.

TABLE 2. Baseline configuration and variations.

Characteristic	Baseline	Variations
Topology (mesh)	4×4	8×8
VCs/VN	2	4
Flit buffers/VC	4	3, 2
SFP ratio	80%	60%, 40%

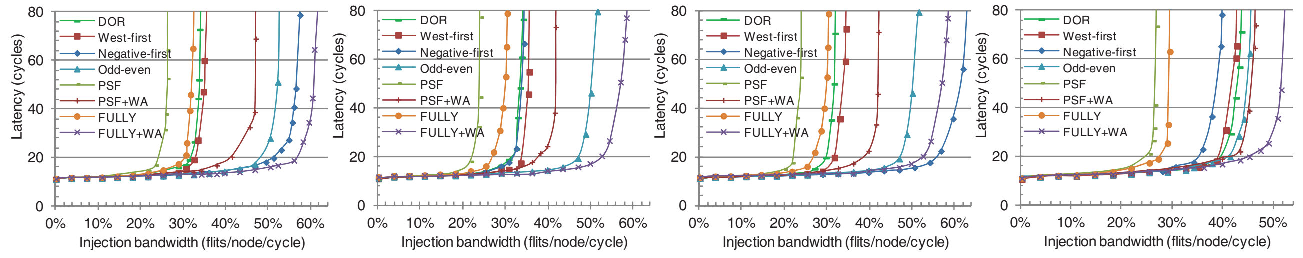
5.1 Single-flit Packet Ratio (SFP ratio)

Single-flit packet (SFP) ratios depend on the cache hierarchy, the coherence protocol and the application. To test the robustness of our design, we evaluate 60% and 40% SFP ratios for transpose-1. As shown in Fig. 6, DOR, west-first, negative-first and odd-even exhibit nearly identical performance for different SFP ratios. The aggressive VC re-allocation makes their performance insensitive to the packet length distribution. However, the performance of PSF and FULLY improves as the SFP ratio shrinks. Their conservative VC re-allocation favors long packets which utilize buffers more efficiently than short ones. As the SFP ratio decreases, so does the possibility of applying WPF. Thus, the performance gap between FULLY+WA and FULLY (or PSF+WA and PSF) decreases. However, even with a 40% SFP ratio, FULLY+WA achieves a 53.1% saturation throughput gain over FULLY.

5.2 VC Depth

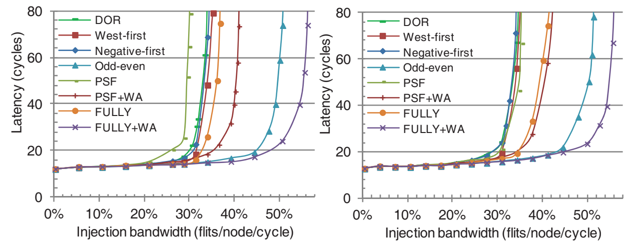
Different NoCs may use different VC depths. To test the flexibility, we evaluate 3- and 2-flit deep VCs with bit reverse traffic. Comparing 4 flits/VC (Fig. 5a) and 3 flits/VC (Fig. 7a), DOR and west-first perform similarly, while FULLY and PSF show minor performance degradation. DOR and west-first offer no or very limited adaptiveness which is a major factor in their performance. Thus, reducing the VC depth from 4 to 3 has little effect. The bottleneck of FULLY and PSF is conservative VC re-allocation. Considering the majority of short packets, reducing the VC depth from 4 to 3 only affects performance slightly. However, the performance of FULLY+WA, PSF+WA, odd-even and negative-first declines with shallower VCs since the VC depth is their bottleneck. Shallow VCs increase the number of hops that a blocked packet spans, which increases the effect of chained blocking [23].

Comparing 3 and 2 flits/VC, performance drops for all algorithms. FULLY outperforms DOR and west-first with 2 flits/VC. As the VC depth decreases, the difference between aggressive and conservative VC re-allocation declines; FULLY gets a relative gain. Even with 2 flits/VC, WPF still optimizes the performance since short packets dominate traffic. In Fig. 7b, FULLY+WA performs 46.2% better than FULLY. Novel flow control leads to superior performance even with



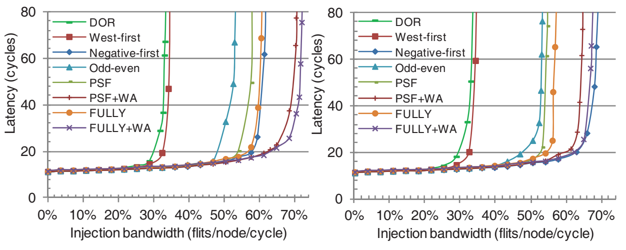
(a) Bit reverse. (b) Transpose-1. (c) Transpose-2. (d) Hotspot.

Fig. 5. Routing algorithm performance for the baseline configuration (A reproduction of Fig. 10 in the main file).



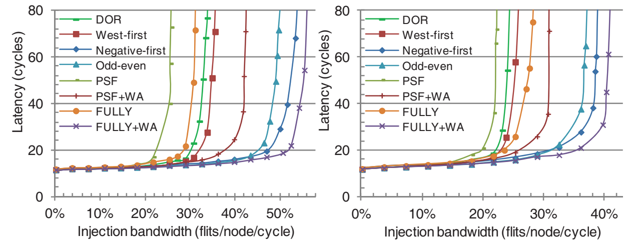
(a) 60% SFP ratio (b) 40% SFP ratio.

Fig. 6. The performance with different SFP ratios for transpose-1 traffic.



(a) Bit reverse. (b) Transpose-2.

Fig. 8. The performance with 4 VCs/VN.



(a) 3-flit deep VCs. (b) 2-flit deep VCs.

Fig. 7. The performance with different VC depths for bit reverse traffic.

half of the buffers, which enables the design of a low-cost NoC. With 2 flits/VC (Fig. 7b), FULLY+WA's saturation throughput is 40.3%, while FULLY's saturation throughput is 32.3% with 4 flits/VC (Fig. 5a). The same is true for PSF+WA.

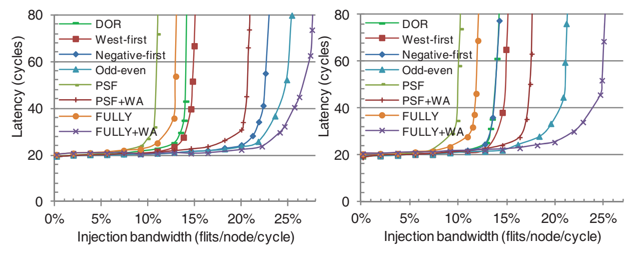
5.3 VC Count

Semiconductor scaling and coherence protocol optimization may allow a VN to be configured with more VCs. Comparing 4 VCs/VN (Fig. 8a) and 2 VCs/VN (Fig. 5a), the performance of DOR, west-first and odd-even is almost the same. These algorithms offer limited adaptiveness; although additional results show increasing the VC count from 1 to 2 improves performance, increasing the VC count from 2 to 4 cannot reduce the physical path congestion and does not further improve performance. Negative-first has a modest gain. In contrast, PSF, FULLY, PSF+WA and FULLY+WA all have significant gains; more VCs mitigate the negative effects of conservative VC reallocation. The gap between PSF and FULLY (or PSF+WA and FULLY+WA) decreases with more VCs; more VCs reduce the possibility of using escape VCs in PSF which forces packets to lose adaptivity.

Fig. 8b shows the performance of transpose-2, which is a favorable pattern for negative-first. FULLY+WA performs similarly to negative-first; with more VCs, the effect of restricting the use of escape VCs in FULLY+WA declines. More VCs reduces the gap between FULLY and FULLY+WA (or PSF and PSF+WA), since the possibility of facing empty VCs increases. Furthermore, using WPF to forward entire packets into non-empty VCs may result in Head-of-Line (HoL) blocking [4] and limit FULLY+WA (or PSF+WA). Nevertheless, FULLY+WA still shows an average 19.8% gain over FULLY for these two patterns with 4 VCs; providing high VC utilization outweighs the negative effect of HoL blocking in a VC-limited NoC. Similar to VC depth, with only 2 VCs, FULLY+WA performs similarly or even better (Figs 5a and 5b) than FULLY with 4 VCs (Fig. 8). WPF provides similar or higher performance with half as many VCs.

5.4 Network Size

Fig. 9 explores the scalability for an 8×8 mesh. The trends across different algorithms are similar to the 4×4 mesh (Fig. 5). Communication is mostly determined by the traffic pattern. Since a larger network leads to higher average hop counts [14], it puts more pressure on VCs than a smaller one. Novel flow control achieves more gains in a larger network. The average improvement for these two patterns of FULLY+WA over FULLY is 108.2%, while it is 93.1% in a 4×4 mesh. As packets travel more hops in a larger network, the possibility of entering an escape VC increases. For PSF and PSF+WA, once the packet enters an escape VC, it loses adaptivity in subsequent hops. Therefore, the gap between FULLY+WA and PSF+WA (or FULLY and PSF) increases with a larger



(a) Bit reverse. (b) Transpose-1.

Fig. 9. The performance for an 8×8 mesh network.

network; providing routing flexibility becomes more important with a larger network.

6 Further Discussion

6.1 Packet Length

Packet lengths for cache coherence traffic typically have a bimodal distribution. However, optimizations such as cache line compression [5], [11] create packet distributions that are not bimodal; the packet length may be distributed between a single flit and the maximum flits per packet supported by the architecture. To apply WPF on such NoCs, more downstream VC status registers are needed for the first-stage arbiters shown in Fig. 9 of the main file. An important consideration is how many different packet lengths to apply WPF to. The longest packet length that can use WPF is one flit shorter than the VC depth. Designers can ignore long packets, since there are few opportunities to apply WPF on long packets. This tradeoff depends on the packet length distribution, VC depth, hardware overhead and the expected performance gain.

6.2 DAMQ and Hybrid Flow Controls

Previous research proposed dynamically allocated multi-queue (DAMQ) designs for both off-chip [22] and on-chip networks [17], [26] to improve VC utilization. Even with DAMQ, allowing multiple packets to reside in one VC may lead to deadlock similar to Fig. 3 of the main file for fully adaptive routing in wormhole networks. WPF is complimentary to DAMQ as it ensures deadlock-freedom. WPF can be viewed as a hybrid mechanism combining wormhole and VCT. There are some hybrid flow controls [13], [19], [20]. Hybrid switching [19] and buffered wormhole [20] remove a blocked packet to release held physical channels by using either the processing node memory [19] or a central buffer [20]. Layered switching divides a long packet into several groups and tries to keep SA grants for a whole group [13]. Our research is different; we focus on improving the performance while avoiding deadlock.

Acknowledgments

We thank the anonymous reviewers for their thorough and constructive suggestions. This work is supported by 863 Program of China (2012AA010905), NSFC (61272144, 61070037, 61272143, 61025009), the University of Toronto, NSERC of Canada, Research Project of NUDT (JC13-06-02).

References

[1] C. Bienia et al. The PARSEC benchmark suite: characterization and architectural implications. In *PACT 2008*.
[2] G. Chiu. The odd-even turn model for adaptive routing. *IEEE Trans. Parallel Distrib. Syst.*, 11(7):729 –738, Jul. 2000.
[3] W. Dally and C. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Trans. Computers*, C-36(5):547 –553, May 1987.
[4] W. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.
[5] R. Das et al. Performance and power optimization through data compression in NoC architectures. In *HPCA 2008*.
[6] J. Duato. A new theory of deadlock-free adaptive routing in wormhole networks. *IEEE Trans. Parallel Distrib. Syst.*, 4(12):1320 –1331, Dec. 1993.
[7] J. Duato. A necessary and sufficient condition for deadlock-free adaptive routing in wormhole networks. *IEEE Trans. Parallel Distrib. Syst.*, 6(10):1055 –1067, Oct. 1995.
[8] E. Fleury and P. Fraigniaud. A general theory for deadlock avoidance in wormhole-routed networks. *IEEE Trans. Parallel Distrib. Syst.*, 9:626–638, Jul. 1998.
[9] B. Fu, Y. Han, J. Ma, H. Li, and X. Li. An abacus turn model for time/space-efficient reconfigurable routing. In *ISCA 2011*.
[10] C. Glass and L. Ni. The turn model for adaptive routing. In *ISCA 1992*.
[11] Y. Jin et al. Adaptive data compression for high-performance low-power on-chip networks. In *MICRO 2008*.
[12] X. Lin, P. McKinley, and L. Ni. The message flow model for routing in wormhole-routed networks. *IEEE Trans. Parallel Distrib. Syst.*, 6(7):755 –760, Jul. 1995.
[13] Z. Lu, M. Liu, and A. Jantsch. Layered switching for networks on chip. In *DAC 2007*.
[14] S. Ma, N. Enright Jerger, and Z. Wang. DBAR: an efficient routing algorithm to support multiple concurrent applications in networks-on-chip. In *ISCA 2011*.
[15] P. Magnusson et al. Simics: A full system simulation platform. *Computer*, 35:50–58, Feb. 2002.
[16] N. Neelakantam et al. FeS2: A full-system execution-driven simulator for x86. In *Poster presented at ASPLOS 2008*, 2008.
[17] C. Nicopoulos et al. ViChaR: A dynamic virtual channel regulator for network-on-chip routers. In *MICRO 2006*.
[18] L. Schwiebert and D. N. Jayasimha. A necessary and sufficient condition for deadlock-free wormhole routing. *J. Parallel Distrib. Comput.*, 32:103–117, Jan. 1996.
[19] K. Shin and S. Daniel. Analysis and implementation of hybrid switching. In *ISCA 1995*.
[20] C. Stunkel et al. The SP2 high-performance switch. *IBM Syst. J.*, 34:185–204, Apr. 1995.
[21] S. Taktak, E. Encrenaz, and J. Desbarbieux. A polynomial algorithm to prove deadlock-freeness of wormhole networks. In *PDP 2010*.
[22] Y. Tamir and G. Frazier. High-performance multiqueue buffers for vlsi communication switches. In *ISCA 1988*.
[23] A. Vaidya, A. Sivasubramaniam, and C. Das. Impact of virtual channels and adaptive routing on application performance. *IEEE Trans. Parallel Distrib. Syst.*, 12(2):223 –237, Feb. 2001.
[24] F. Verbeek and J. Schmaltz. Automatic verification for deadlock in networks-on-chips with adaptive routing and wormhole switching. In *NoCS 2011*.
[25] F. Verbeek and J. Schmaltz. On necessary and sufficient conditions for deadlock-free routing in wormhole networks. *IEEE Trans. Parallel Distrib. Syst.*, 22(12):2022 –2032, Dec. 2011.
[26] Y. Xu et al. Simple virtual channel allocation for high throughput and high frequency on-chip routers. In *HPCA 2010*.