2011 International Conference on Parallel Architectures and Compilation Techniques

Exploiting Mutual Awareness between Prefetchers and On-chip Networks in Multi-cores

Junghoon Lee, Minjeong Shin, Hanjoon Kim, John Kim, Jaehyuk Huh
Department of Computer Science, KAIST
{ijij41,shinmj,hanj,jjk12,jhhuh}@kaist.ac.kr

***Abstract*—The unique characteristics of prefetch traffic have not been considered in on-chip network design for multi-core architectures. Most prefetchers are often oblivious to the network congestion when generating prefetech requests. In this work, we investigate the interaction between prefetchers and on-chip networks and exploit the synergy of these two components in multi-core architectures. We explore prefetch-aware on-chip networks that differentiates between prefetch and demand traffic by prioritizing demand traffic. In addition, we propose prefetch control mechanism based on network congestion. Our evaluations show that the combination of the proposed prefetch-aware router architecture and congestion-sensitive prefetch control improves the performance of benchmarks by 11-13% on average, up to 30% on some of the workloads.**

I. Introduction

A hardware prefetching techniques have been widely used in microprocessors to hide the long memory latencies [3]. With the migration towards multi-core architectures, prefetch traffic needs to traverse the on-chip network that connects all the on-chip components together [2]. However, design of on-chip networks have not considered the unique characteristics of network traffic generated by prefetchers. Unlike request and reply (data) traffic, prefetch traffic is essentially speculative, and thus can be useless, if the prediction is incorrect. Even for successfully predicted prefetch requests, the data may be actually used hundreds or thousands cycles after the initiation of prefetch requests. In addition to the prefetch oblivious network design, most prefetchers are also insensitive to the status of networks. As shown in Figure 1, prefetch traffic accounts for a significant portion of the overall traffic on on-chip networks. [1] This indicates that networks must be aware of the prefetch traffic, which have different requirements for criticality from the demand traffic. As the amount of prefetch traffic is significant, the opportunity arising from the distinct characteristics of prefetch traffic should be exploited.

In this work, we investigate how the two components, on-chip networks and hardware prefetchers, affect each other,

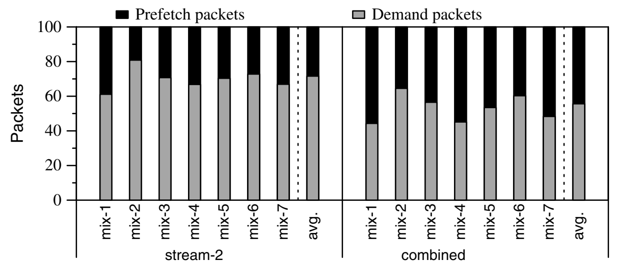

Figure 1. The decomposition of the total packets into demand and prefetch packets

and explore the design space of mutually aware networks and prefetchers. We first study the network design for two distinct type of traffics generated by the cores and the prefetchers. We propose a priority-based arbitration mechanism for routers, called *Traffic-aware Prioritized Arbiter (TPA)*, which gives a higher priority to non-prefetch packets than prefetch packets. In addition, we propose a prefetch control mechanism, which adjusts the prefetch traffic based on the status of networks, called *Traffic-aware Prefetch Throttling (TPT)*. TPT throttles prefetch generations at the hardware prefetchers, depending on network congestion.

II. Prefetch-Aware On-Chip Network

Since prefetch packets should have less priority than demand packets, we evaluate alternative prefetch-aware on-chip networks which separate network resources according to their priority. The two main network resources in on-chip networks are the buffers and the channel bandwidth. *Partitioned VC* is on network separates the buffer resources (or VCs) between the two type of traffic. Another alternative is *partitioned network* where the channel bandwidth is partitioned into multiple channels to create multiple networks in parallel – similar to channel slicing [2] but the network is selected based on the traffic type.

In addition to partitioning network resources, we propose traffic-aware prioritized arbitration (TPA) which does not require separating any network resources. With TPA, round-robin arbitration is first done among all the demand packets. If there are no demand packets, then round-robin arbitration is done among the prefetch packets. To prevent priority inversion (i.e., demand packets being buffered behind prefetch packets), we only allocate a VC to a new packet once

[1]For our evaluation in this work, we use the Simics with the GEMS and Garnet network model (64 in-order cores with 32KB L1, shared 64 L2 256KB banks, 8x8 concentrate mesh (c-mesh) [1] networks), and a stream prefetcher with a degree of two (`stream-2`) and a combined prefetcher which adds the next-line prefetching to the stream-2 prefetcher (`combined`).

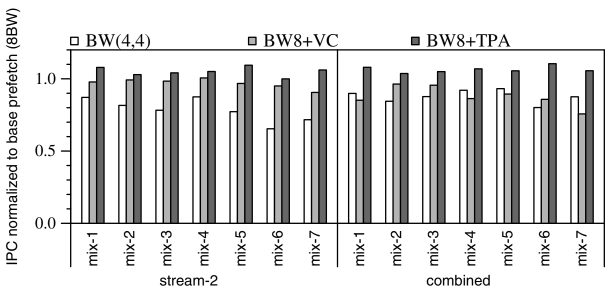

Figure 2. Performance comparison of alternative prefetch-aware on-chip network architecture.

the previous packet has departed the downstream router. With fixed prioritized arbitration, the starvation of prefetch packets can occurs. To prevent this, if a prefetch packet has not been serviced for t cycles, the prefetch packet is *upgraded* and receive priority.

In Figure 2, we compare the performance against a baseline on-chip network with prefetching enabled and compare the following alternatives – BW(4,4), a partitioned network where the channel bandwidth of 8 bytes is partitioned into two networks of 4 bytes, one network for prefetch and another network for demand traffic; BW8+VC where the VC is partitioned into separate VCs for prefetch and demand packets; and BW8+TPA where the prioritized arbitration is used but no resource partitioning. Both resource partitioning (BW(4,4) and BW8+VC) result in poor performance, compared with the baseline; in particular, BW(4,4) can result in up to 35% loss in performance compared with the baseline. In comparison, BW8+RAP shows improvement in performance, by up to 10%. Thus, despite the different characteristics of demand and prefetch traffic, segregating network resources (i.e., router buffers and network bandwidth) does not result in performance improvement as the shared resource is partitioned statically among the different traffic. However, by using prioritized arbitration, traffic can share all the network resources and result in performance improvement, with minimal impact on router complexity.

III. Network-Aware Prefetchers

To complement prefetch-aware network, we also propose traffic-aware prefetch throttling (TPT) that detects network congestion dynamically and throttles prefetch requests if the network is congested. TPT tracks congestion in the network and if the generated prefetch request needs to be routed through one of the congested paths, the request is throttled. A *congested path* from a source to a destination is defined as a path in the network which has longer packet latency than the unloaded (i.e. zero-load) latency from the same source to the destination. We assume each packet carries a timestamp to estimate path congestion. The congestion information is returned by piggybacking on packets returning from the destination back to the source. No centralized resource is required as TPT is distributed with each node tracking congestion and determining whether throttling is needed.

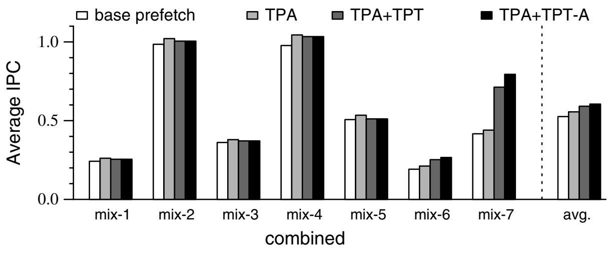

Figure 3. Performance with network-aware prefetchers

In addition to the network status, the accuracy of prefetching also helps determine the usefulness of prefetch traffic and has been previously proposed [4]. The TPT can be augmented with prefetch accuracy information in throttling prefetching requests. Thus, we propose TPT with accuracy-awareness (TPT-A). Different implementation of TPT-A is possible which includes combining the network congestion and prefetch accuracy information together to determine a threshold for throttling. In our initial implementation of TPT-A, when prefetch requests are throttled based on network congestion, the prefetch requests are still injected *if* the request is determined to have a very high accuracy.

Figure 3 presents results for TPA-only, TPA+TPT, and TPA+TPT-A configurations – compared with baseline prefetcher. TPT improves performance by up to 30% with an aggressive combined prefetcher. For workloads with relatively low prefetch generation rates, the congestion level is relatively low and very few prefetch requests are throttled. TPT-A improves TPT in general as highly accurate prefetches are still injected into the network and help improve overall performance. As a result, our initial results indicate that aggressive prefetchers must be sensitive to network traffic. In addition, combination of both TPA and TPT can help increase overall performance by exploiting the mutual awareness between the prefetcher and the on-chip network.

Acknowledgment

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2011-0015039).

References

[1] J. Balfour and W. J. Dally. Design tradeoffs for tiled cmp on-chip networks. In *ICS '06: Proceedings of the 20th annual international conference on Supercomputing*, pages 187–198, New York, NY, USA, 2006. ACM.

[2] W. J. Dally and B. Towles. Route packets, not wires: on-chip interconnection networks. In *Proceedings of the Design Automation Conference*, pages 684–689, Las Vegas, NV, June 2001.

[3] E. Ebrahimi, C. J. Lee, O. Mutlu, and Y. N. Patt. Prefetch-aware shared resource management for multi-core systems. In *ISCA*, pages 141–152, 2011.

[4] E. Ebrahimi, O. Mutlu, C. J. Lee, and Y. N. Patt. Coordinated control of multiple prefetchers in multi-core systems. In *MICRO 42: Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, pages 316–326, New York, NY, USA, 2009. ACM.