2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture

STEM: Spatiotemporal Management of Capacity for Intra-Core Last Level Caches

Dongyuan Zhan, Hong Jiang, Sharad C. Seth
Dept. of Computer Science & Engineering
University of Nebraska - Lincoln
Email: {dzhan,jiang,seth}@cse.unl.edu

***Abstract*—Efficient management of last level caches (LLCs) plays an important role in bridging the performance gap between processor cores and main memory. This paper is motivated by two key observations, based on our study of LLCs: 1) the capacity demand is highly non-uniform and dynamic at the set level; and 2) neither spatial nor temporal LLC management schemes, working separately as in prior work, can consistently and robustly deliver the best performance under different circumstances. Therefore, we propose a novel adaptive scheme, called STEM, which concurrently and dynamically manages both spatial and temporal dimensions of capacity demands at the set level. In the proposed scheme, a set-level monitor captures the temporal and spatial capacity demands of individual working sets and judiciously pairs off sets with complementary capacity demands so that the underutilized set in each pair can cooperatively cache the other's victim blocks. The controller also decides on the best temporal sharing patterns for the coupled sets in the event of inter-set space sharing. Further, if the LLC controller cannot find a complementary set for a particular set, STEM can still decide on the best set-level replacement policy for it. Our extensive execution-driven simulation data shows that the proposed scheme performs robustly and consistently well under various conditions.**

***Keywords*-Chip Multiprocessors; Last Level Cache Management; Set-Level Non-Uniformity of Capacity Demands; Cooperative Caching;**

1 Introduction

As the memory wall continues to limit processors' performance, judiciously managing on-chip Last Level Caches (LLC) has become increasingly critical to bridging the speed gap between processor cores and memory subsystems. Because of LLCs' vital importance to the overall system performance, LLC management schemes have been studied extensively in [1–7]. In particular, previous work has shown that the traditional LRU replacement policy cannot be optimal when workloads exhibit poor temporal cache locality. Several alternative policies, such as DIP [1] and PeLIFO [2], have been proposed to improve LLCs' performance by employing sophisticated block insertion, promotion, and victimization strategies. Moreover, researchers have observed that, independent of the replacement policy, LLCs can exhibit very distinct resource demands at the set level because of the non-uniform characteristics of working sets that are mapped to individual LLC sets. As a result, several recent proposals, such as V-Way [3] and SBC [4], by aiming to provide better cooperation between LLC sets in retaining working sets, could outperform the aforementioned replacement policies under certain circumstances.

It is our contention that the two kinds of cache management schemes, mentioned above, are inherently different in that one is rooted in the temporal management and the other in the spatial management of LLC capacity resources. Specifically, we define *temporal resource management* as replacement policies (such as DIP and PeLIFO) that determine how the capacity of an LLC set is temporally shared among the competing blocks of a working set mapped to the LLC set, when the LLC set cannot retain all of them. Furthermore, we define *spatial resource management* as schemes (such as V-Way and SBC) that dynamically decides how the overall capacity of an LLC is spatially partitioned among LLC sets that are hosting different working sets. Our execution-driven simulation demonstrates that neither the temporal nor the spatial LLC management schemes, working independently, can consistently and robustly deliver the best performance in all situations. To better understand the differences between the two dimensions of management, we characterize the non-uniform distribution of working sets' spatial and temporal capacity demands and its performance impact, and conclude that the effectiveness of a specific LLC management strategy is determined by how an LLC's set-level capacity provision and utilization meet its non-uniform set-level capacity needs.

Motivated by the observations on the philosophical gap between existing spatial and temporal LLC management schemes as well as its significant performance impact, we propose the adaptive *SpatioTEmporal Management* (STEM) scheme to regulate the two dimensions of capacity demands concurrently and dynamically. In the proposed scheme, a set-level monitor based on shadow-tag hash signatures and saturating counters is utilized to capture and measure both temporal and spatial capacity needs of individual working sets. Based on these measurements, the cache controller then judiciously identifies and pairs off sets with a complementary capacity demand, thus enabling the underutilized set in each pair to cooperatively cache the other's victim blocks while deciding the best temporal sharing patterns for both of them in the event of inter-set space sharing/inter-set

1072-4451/10 $26.00 © 2010 IEEE
DOI 10.1109/MICRO.2010.31

cooperative caching. In addition, if a set cannot find another set with a complementary capacity demand to pair with, the controller can still decide the best set-level replacement policy for it.

Our execution-driven simulation using 15 benchmarks shows that the proposed scheme performs robustly and consistently well under various workloads and HW configurations studied. Specifically, our STEM LLC design can improve the metrics of misses per 1k instruction (MPKI), average memory access time (AMAT) and cycle per instruction (CPI) by 21.4%, 13.5% and 6.3% over LRU respectively, which is better than the state-of-the-art DIP, PeLIFO, V-Way and SBC schemes, at a manageable HW storage cost of only 3.1%.

The main contributions of this work are:

- The explicit classification of temporal and spatial LLC capacity management, unique analyses of the distinct working principles and comfort zones of the two dimensions, and key observations on their impacts on the performance variations of state-of-the-art LLC management schemes.
- A novel LLC design called STEM that judiciously adapts the cache management strategy to capacity demands in both spatial and temporal dimensions concurrently and dynamically.
- The thorough evaluation and key conclusions through extensive execution-driven simulation.

The rest of this paper is organized as follows. Section 2 introduces the background of conventional LLC organization and management, followed by the formulation of our research problem. Section 3 demonstrates our research motivation. Section 4 elaborates on the design issues of our proposed STEM scheme. Section 5 shows the experiment setup used for evaluation and provides an analysis of the obtained results. Related work is discussed in Section 6 and the paper concludes with a summary in Section 7.

2 Background

In this section, we first provide the necessary background for the organization and management of conventional set-associative LLCs, which motivates us to view the conventional problems from an unconventional perspective.

2.1 Conventional LLC Organizations

On-chip LLCs are utilized to retain as many blocks on-chip as possible for near future reuse. An on-chip cache is typically organized in three tiers internally. The 1^{st} tier is the cache entity itself to which the upper-level memory hierarchy components send requests. E.g., an L2 cache receives references from L1 caches. In the 2^{nd} tier, a cache is organized in sets for the simplicity of address decoding which essentially divides the cache's overall access stream into a number of subsequences, namely, working sets, which are mapped to individual cache sets. In practice, the ordinary MOD function is the simplest and widely adopted mapping scheme, with the modulo base equal to the number of cache sets (typically an integral power of 2). Then, the accesses whose target addresses have the same congruence relation will be mapped to the same cache set and thus form a working set. The 3^{rd} tier is the cache line, which is the basic unit of resource management in the cache organization. All cache lines assigned to the same set will be used to host the member blocks of the corresponding working set, and the number of lines in a set is defined as the set's associativity. Also for simplicity, all cache sets have the same static associativity in a conventional organization.

2.2 The Problems of Conventional LLC Management

First, as observed in recent studies [3, 4, 8], an interesting LLC property known as the set-level non-uniformity of capacity demands can result in the underutilization of those LLC sets whose working sets require less than the associativity, while leaving other sets overutilized because their capacity (i.e., associativity) is insufficient for their working sets. Therefore, state-of-the-art LLC spatial management schemes such as V-way and SBC attempt to perform dynamic cache line assignment to different LLC sets according to their spatial metrics. For the V-way cache, the metric is implicitly the per-set "access count", while SBC's metric is the "saturation level" defined as the difference between the miss and hit counts at the set level.

Second, when a working set cannot be entirelly retained in a cache set, its member blocks will compete for the set's cache lines, giving rise to policies that decide which block needs be evicted from a set in the event of a block replacement. Existing HW-replacement policies all use certain criteria to adjust the lifetime values of cached and incoming blocks so as to approximate the ideal "Belady's optimal algorithm" [9]. Such criteria can make a significant difference in the LLC performance. For instance, the simple and commonly used LRU replacement policy favors access (both hit and miss) recency when adjusting blocks' lifetime in caches. Therefore, it performs quite well when a working set exhibits excellent temporal locality but can thrash an LLC set when the locality is poor. The more advanced DIP replacement policy always advocates hit recency but duels between either favoring or penalizing miss recency (namely, assigning the recently-missed/incoming block with either the longest or shortest lifetime). As a result, DIP is more flexible and adaptive than LRU in making replacement decisions.

2.3 Unconventional Thinking of the Problems

From the analysis above, we argue that the two different types of approaches actually have fundamentally distinct working principles, of which one is to spatially manage LLC capacity partition among different LLC sets (such as V-Way and SBC) and the other is to temporally optimize the sharing pattern of an LLC set's capacity among its working

set's member blocks. More specifically, if an overutilized LLC set can get sufficient cooperative capacity from another underutilized set to retain both of their working sets, no replacement needs to take place. In this situation, the spatial management schemes will obviously be more effective than the temporal approaches that manage the two sets separately. On the other hand, if an LLC set does not have enough local space for its working set or cannot find external capacity for cooperation, then adopting an advanced replacement policy such as DIP will be more sensible. But one of the challenges here is that existing adaptive temporal approaches such as DIP and PeLIFO all depend on application/LLC-level sampling, monitoring and decision-making, rendering them unable to work on an individual set basis. Yet, working at the set level, we believe, is essential in addressing the issue of set-level uniformity of capacity demands, as will be shown in Section 5.2. More challenging is the fact that, if a set can only find some but insufficient cooperative capacity for the additional requests of its working set, both spatial and temporal management should simultaneously take effect on the set and its cooperative set to make the best spatial and temporal use of their aggregate capacity.

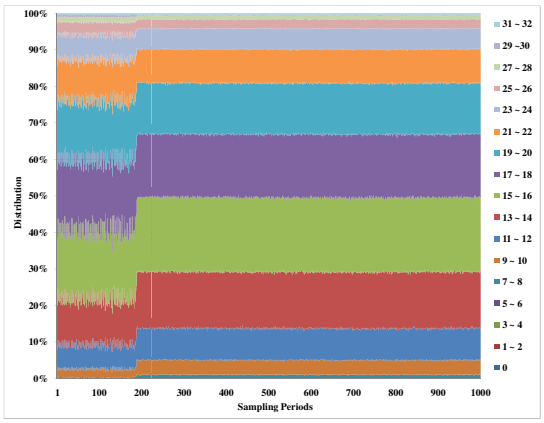
3 Quantitative Motivation

In this section, we study workload characteristics to show the widespread existence of set-level non-uniformity of capacity demands and demonstrate its performance impacts on LLC management schemes, by using intuitive synthetic workloads as well as real-world applications.

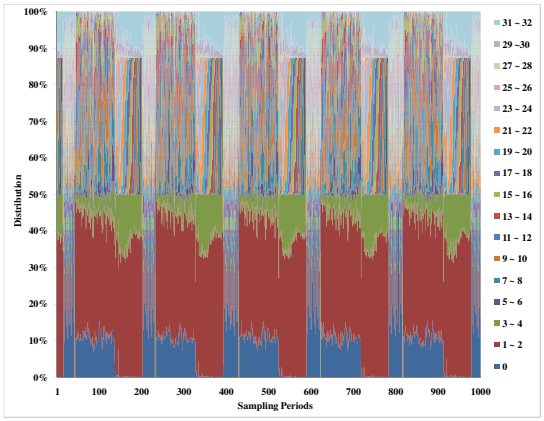
3.1 Non-Uniform Set-Level Capacity Demands

Although the non-uniformity of set-level accesses and saturation levels (defined as the difference between set-level hit and miss counts) has been noted, respectively, in V-Way [3] and SBC [4], we argue that neither the "access count" nor the "saturation level" is an accurate measure of set-level capacity demands. For example, if a set is experiencing a number of accesses, and further if these accesses only touch a small working set, it is highly likely that all accesses eventually turn out to be hits and the working set can be retained in the set without the need for extra capacity. Thus, a high level of access counts is not always indicative of extra capacity demands. On the other hand, an LLC set with misses dominating, may not benefit from receiving extra capacity at all if its working set is of streaming features, while nominally an underutilized set with 80% accesses, e.g., as hits, may be able to further resolve its remaining 20% missed accesses by receiving a small amount of extra capacity.

We use a more accurate approach similar to the one in [8] to characterize the set-level capacity demands of a workload, which essentially defines the capacity demand of a set during a time interval as the minimum number of cache lines required to resolve all conflict misses of the set. We experiment



(a) *omnetpp*



(b) *ammp*

Figure 1. **Distribution of the set-level capacity demands for *omnetpp* and *ammp* during 1000 sampling periods:** each color represents 2 cache ways in the associativity range, according ot the legend shown on the right. E.g., for *omnetpp*, the "light green" band (corresponding to legend "15-16") indicates that about 20% of the sets require 15-16 cache lines per set to meet their capacity demands; for *ammp*, the "blue" band (corresponding to legend "0") reveals that the corresponding sets are streaming-like and thus require almost no capacity.

on two representative benchmarks *omnetpp* and *ammp* to characterize the features of their set-level capacity demands, as illustrated in Figure 1(a) and Figure 1(b) respectively. The detailed description of the experimental setup appears in Section 5. Here, we only list the most important four parameters: 2048 LLC (L2) sets; 64-byte cache lines; 50000 accesses per time interval; and a total of 1000 time intervals during the workload characterization. With the settings, we first identify that the entire application/LLC-level capacity demands are no greater than 32 ways in both cases, which means that an associativity of 32 can help the LLC resolve all conflict misses for the workloads. Then, for an LLC set, we obtain the minimum number of ways/blocks required by it to resolve as many conflict misses as with an associaitivity of 32, and then define the value as the set's current capacity demand. As illustrated in Figure 1, the non-uniformity of set-level capacity demands is very evident for both benchmarks:

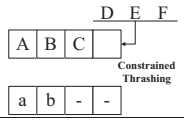
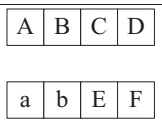
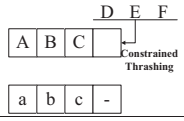
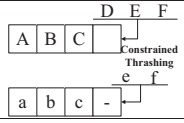
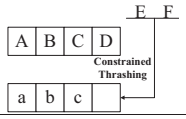
<table>
<tr><th>Ex. #</th><th>Synthetic Workloads</th><th>LRU</th><th>DIP</th><th>SBC</th></tr>
<tr><td rowspan="2">1</td><td rowspan="2">A→a→B→b→C
→a→D→b→…</td><td>The entire working set 0 is thrashing, while working set 1 is well retained in LLC set 1.</td><td>  </td><td>  </td></tr>
<tr><td>Miss Rate = 1/2</td><td>Miss Rate = 1/4</td><td>Miss Rate = 0</td></tr>
<tr><td rowspan="2">2</td><td rowspan="2">A→a→B→b→C
→c→D→a→E→
b→F→c→A→…</td><td>Working set 0 is thrashing, while working set 1 is well retained in LLC set 1.</td><td>  </td><td>It is a dynamic process, and we write a simple trace simulator by following the procedure in the SBC proposal to get the miss rate.</td></tr>
<tr><td>Miss Rate = 1/2</td><td>Miss Rate = 1/4</td><td>Miss Rate = 1/3</td></tr>
<tr><td rowspan="2">3</td><td rowspan="2">A→a→B→b→C
→c→D→d→E→e
→F→a→A→b→…</td><td>Working set 0 is thrashing, and so is working set 1.</td><td>  </td><td>It behaves exactly the same as LRU, due to the absence of underutilized LLC sets.</td></tr>
<tr><td>Miss Rate = 1</td><td>Miss Rate = 1/4+1/5</td><td>Miss Rate = 1</td></tr>
<tr><td colspan="2" rowspan="2">An Extensional Example for Ex. #2</td><td>  </td><td colspan="2" rowspan="2">Still for Ex. #2, if SBC was able to combine its spatial management capability with a more advanced temporal scheme, e.g., retaining “D” in set 0’s local capacity and let E and F compete for the cooperative capacity in set 1, the overall miss rate would be reduced from 1/3 to no greater than 1/6.</td></tr>
<tr><td>Miss Rate ≤ 1/6</td></tr>
</table>

Figure 2. **Conceptual illustration with synthetic workloads.** The construction of synthetic workloads is detailed as follows: 1) in Example #1, the cyclic working Set 0 is “A→B→· · ·→F→A→B→· · ·”, while working Set 1 is “a→b→a→b→· · ·”; 2) in Example #2, the only difference from Example #1 is that working Set 1 has an additional element “c”; 3) in Example #3, working Set 1 has two more elements “d” and “e” than that in Example #2.

for *omnetpp*, almost 50% sets require no more than 16 cache lines per set, while for ammp, about 50% sets require no more than 4 cache line per set.

Next, we show the impact of the non-uniformly distributed set-level capacity demands on LLC management schemes, first for synthetic workloads and then for real applications.

3.2 Demonstration with Synthetic Workloads

For an intuitive illustration, we assume a simple 4-way associative LLC with just two sets. The LLC receives a sequence of repetitive requests from the upper level memory hierarchy components. After mapping the reference sequence to individual LLC sets, we can obtain two cyclic working sets, as shown in Figure 2. For the resulting performance, we measure the LLC’s miss rate after its initialization. We consider SBC and DIP as representatives, respectively, as examples of spatial and temporal LLC management schemes. Furthermore, we assume that DIP has the knowledge of working sets’ patterns without the need for dedicated sampling and dueling monitors [1].

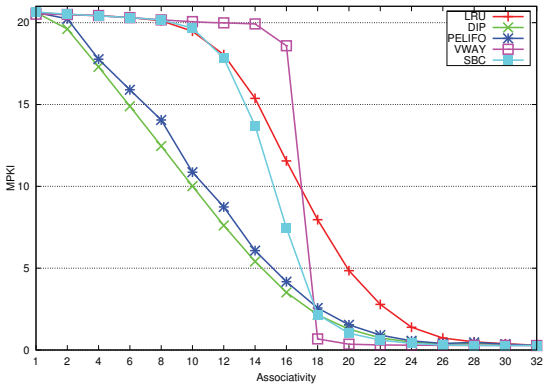
As illustrated in Example #1 of Figure 2, we find that SBC performs better than DIP because SBC enables the overutilized LLC Set 0 to place its blocks in Set 1, and this perfect match does not even trigger any replacements in the long run. In Example #2, although in SBC Set 0 is able to utilize the cooperative (albeit, insufficient) space of Set 1, their underlying LRU replacement policies cannot help the two sets produce the best performance. In Example #3, both LLC sets are overutilized, leaving SBC no choice for inter-set cooperation but to thrash both LLC sets. DIP can keep part of the working sets in both LLC sets, though Set 0 and Set 1 still contribute 1/4 and 1/5 overall misses respectively.

The intuitive illustration above enables us to better understand the different properties and comfort zones between temporal and spatial LLC management schemes. It also reveals that if a spatial management strategy like SBC could incorporate a more advanced temporal management mechanism, a better performance over both spatial and temporal schemes would be achievable. In particular, however, while SBC’s “saturation level” metric works well in the examples of Figure 2, the metric will be shown less effective in the thorough evaluation in Section 5.2.

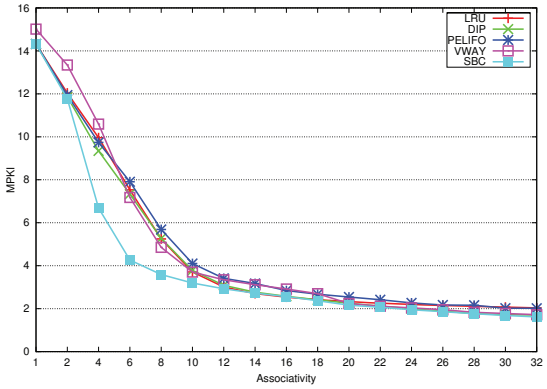
3.3 Demonstration with Real Workloads

We still use *ammp* and *omnetpp* as the representative workloads to evaluate LRU, DIP, PeLIFO, V-WAY and SBC in terms of their misses per 1k instructions (MPKI) under various associativity configurations, as shown in Figure 3.

In Figure 3(a), from associativity 2 to 16, both temporal schemes DIP and PeLIFO outperform both spatial schemes V-Way and SBC, as well as the baseline LRU, for *omnetpp*. From associativity 12 on, the best spatial scheme SBC begins to outperform LRU. From associativity 18 to 24, both spatial schemes perform the best among all schemes.



(a) *omnetpp*



(b) *ammp*

Figure 3. **The MPKIs of *omnetpp* and *ammp* for Different Associativity Configurations**

Beyond associativity 24, there is little difference among the five schemes. SBC's identical performance to LRU when the associativity is less than 12 is consistent with the conclusion drawn from Example #3 in Figure 2 because few sets that are less saturated [4] can be found for an associativity lower than 12. From associativity 12 to 16, SBC's performance is better than LRU but still worse than DIP/PeLIFO. This is because there are some but insufficient less-saturated sets for spatial cooperation in this range, SBC is not able to best utilize the limited cooperative capacity, which is consistent with the conclusion drawn from Example #2 in Figure 2. When the associativity is greater than 18, SBC and V-Way perform the best, because there are an appropriate number of less-saturated sets for cooperation, which is consistent with the conclusion drawn from Example #1 in Figure 2. Beyond associativity 24, the performances of all schemes begin to converge as expected.

In Figure 3(b), from associativity 2 to 10, the best spatial scheme SBC outperforms any temporal schemes for *ammp*. That is because about 50% LLC sets require no more than 4 cache lines per set, as demonstrated in Figure 1(b). Therefore, in the range [4, 10], the spatial scheme is the most effective, which is consistent with the conclusion drawn from Example #1 in Figure 2. When the capacity is beyond

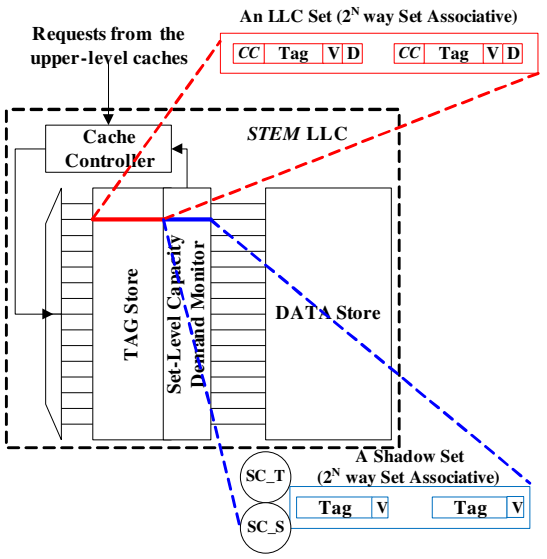


Figure 4. **The STEM Architecture**

10, the effectiveness of both temporal and spatial LLC managements diminishes, because the local/native capacity of each LLC set is sufficient for them to retain their working sets. That is why no other schemes significantly improve over LRU for *ammp* in the associativity range [12, 32].

4 Design & Implementation

It has been clearly illustrated in the motivational experiments that LLCs can exhibit non-uniform capacity demands in both spatial and temporal dimentions. The spatial capacity demands refer to if a working set can fit most of its blocks into the current space of its LLC set, while the temporal capacity demands imply whether the working set is making the best use of the cache space it possesses. The two types of capacity demands have different kinds of impact on the effectiveness of LLC management schemes. That is the principal reason why none of the existing cache management schemes working in either dimension alone can perform robustly and constantly well under all circumstances. Therefore, an adaptive LLC management is required to harness both dimensions of capacity demands concurrently and dynamically to optimize LLCs' performance.

4.1 The STEM LLC Architecture

To accomplish this objective, we propose a novel LLC design named *SpatioTEmporally Managed Last Level Caches* (STEM LLCs). STEM aims to achieve three specific design goals: (1) it identifies the spatial capacity demands of individual sets and couples two sets with complementary needs to perform inter-set cooperative caching; (2) in the event of inter-set space sharing, it determines the best temporal capacity sharing patterns for both of the coupled sets so as to optimally utilize both local and cooperative capacity; and (3) for those uncoupled sets, it is still able to decide the best set-level replacement policy for them.

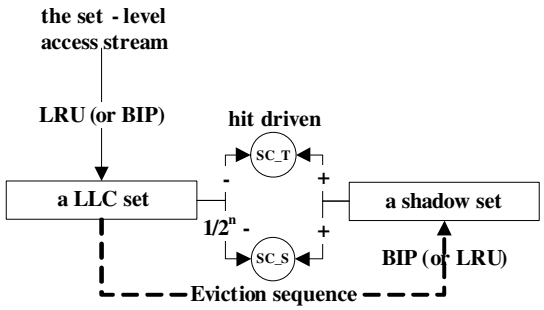


Figure 5. **A Set-Level Capacity Demand Monitor (SCDM)**

Figure 4 provides an architectural view of the STEM LLC. The STEM cache controller accepts access requests from upper-level caches. Then, the controller looks up the referenced block in the tag store, which is decoupled from the data store, to see if it is present in the LLC. There can be two scenarios if the requested block is on-chip: the block is either in its local set with the same index as indicated in the physical address of the block, or in a different set where the block is cooperatively cached. Therefore, each tag store entry needs an additional bit called the CC bit to indicate whether the block is local ($CC = 0$) or cooperatively cached ($CC = 1$), as shown in Figure 4. Then, the requested block is forwarded to the upper-level cache if it is found on-chip or otherwise fetched from DRAM. Meanwhile, the Set-level Capacity Demand Monitor (SCDM) is operated to capture and measure the dynamic information of individual sets' spatial and temporal capacity demands and feed it back to the cache controller. Based on the feedback information, the controller couples two sets with complementary spatial capacity needs and decides their best temporal capacity sharing behaviors for inter-set cooperative caching. For an uncoupled set, STEM will also adapt the set's replacement policy to either LRU or BIP (Binomial Insertion Policy) [1]. The design details and working principles of each critical component in STEM will be elaborated in the following subsections.

4.2 Set-Level Capacity Demand Monitor

The Set-level Capacity Demand Monitor (SCDM) is devised to capture and monitor both spatial and temporal capacity demands of individual sets. Associated with each LLC set, as illustrated in Figure 5, there is a shadow set [10] and two k-bit saturating counters "SC_S" and "SC_T" in the SCDM. Each shadow set has the same associativity as the corresponding LLC set and stores an m-bit hash value taken from the tag field of a victim block from the LLC set, where m is much shorter than the length of a tag field. In the context of this paper, we still call this hashed tag value as a shadow tag. Thus, an LLC set appears to have "double" capacity with the additional "virtual" space provided by its shadow set. Then, the two saturating counters measure the spatial and temporal capacity demands of each LLC set by using the information embodied in the shadow tags.

4.3 Operations on Shadow Sets

There are three essential operations on a shadow set: (1) if a local block is evicted from its original LLC set, the hash value of its tag field will be calculated by STEM's hashing module and inserted into the corresponding shadow set; (2) the shadow set maintains its own independent ranking for all of its valid entries and uses it for replacement; (3) if an access on a local block is missed in an LLC set, the corresponding shadow set will be looked up to check if the tag of the requested block is present in a valid shadow set entry. Additionally, it is required that the shadow set entries be strictly exclusive with the local blocks in the corresponding LLC set in terms of the complete/hash values of the tag fields. Therefore, if a previously evicted block with its tag present in the shadow set is revisited by the owner set, two operations must be performed: (1) the shadow entry that has the hashed tag needs to be invalidated after the corresponding block is inserted into the LLC set; (2) a hit on the shadow set is signaled to operate its saturating counters.

The information of an LLC set's spatial capacity demands can be naturally captured by the shadow set because it contains the information of the set's victim blocks, as shown in Figure 5. If there are a considerable number of hits on the shadow set, it implies that the blocks previously evicted from the LLC set will soon be revisited and extending the LLC set's space will be beneficial. In the STEM LLC design, on the other hand, the shadow set adopts a replacement policy opposite to that of the corresponding LLC set to capture the information of the LLC set's temporal capacity demands. Specifically, as illustrated in Figure 5, if the LLC set is currently adopting the LRU replacement policy to favor temporal locality, the shadow set will use the BIP policy [1] to keep the shadow tags of LLC victim blocks. The rationale behind the specific design choice is that if a large working set cannot be well retained in the LLC set due to its poor temporal locality, e.g., by way of thrashing, then the same information of poor temporal locality will also be reflected by its eviction stream, which in turn can be captured by adopting BIP in the shadow set that contains the information of the set's victim blocks. On the contrary, if an LLC set is adopting BIP for insertion but actually its large working set shows good temporal locality (e.g., if the average reuse distance is shorter than the set associativity), the temporal locality information can be captured in the eviction stream as well if the shadow set takes the LRU replacement policy.

4.4 Operations on Saturating Counters

The two k-bit saturating counters "SC_S" and "SC_T" are used to measure a set's temporal and spatial capacity demand by comparing the hit count of a shadow set against that of the LLC set respectively. Whenever there is a hit on the

shadow set, both saturating counters will be incremented by one. The temporal saturating counter is always decremented by one upon a hit on the LLC set, while the spatial saturating counter is decremented by one for every 2^n hits on the LLC set, as demonstrated in Figure 5. We implement counting of 2^n hits on the LLC set in a probabilistic way that the spatial saturating counter is decremented by one only when an n-bit value produced by a *random number generator* is zero. The random number generator can be simply incorporated in the LLC controller.

We look at the values of the two k-bit saturating counters of an LLC set to measure its spatial and temporal capacity demands. Specifically, if a spatial saturating counter reaches a saturated value, it implies that providing the LLC set with double capacity can result in at least $\frac{1}{2^n}$ increase in the hit rate. The LLC set should be identified as a taker set that can benefit from inter-set cooperation; otherwise, if the MSB (*most significant bit*) of the spatial saturating counter is 0, it suggests that the LLC set has a very high hit frequency in its local capacity and it could be regarded as a giver set that can contribute part of its capacity in inter-set space sharing. The spatial saturting counter is reset only on system initialization. On the other hand, for a temporal saturating counter, if it reaches a saturation value, it indicates that the shadow set's replacement policy is measured to perform better than the LLC set's current policy, which will send a request to the cache controller to swap the replacement policies for the LLC and shadow sets as well as resetting the temporal saturating counter.

4.5 Coupling Sets with Complementary Capacity Demands

As described above, a saturated spatial saturating counter indicates that extending the capacity of the corresponding set is beneficial; hence the set is regarded as a taker set that can significantly reduce its conflict misses when its capacity is extended. On the other hand, a 0-valued MSB denotes a giver set that may need less blocks than it currently possesses. Thus, the STEM LLC should couple a taker set and a giver set so that the taker set can utilize part of the giver set's capacity to reduce conflict misses.

The coupling process needs the assistance of a hardware heap (similar to the Destination Set Selector in [4]) that keeps track of a small number of uncoupled giver sets that are less saturated than others, as well as an association table [4] that maintains the association information of paired sets. If a set is not paired with any other set, the value of its association table entry is the set's own index. Both the HW heap and association table are embedded in the STEM LLC controller. When a set is identified as a giver set by its monitor, it tries to post its index and saturating level information to the heap. The heap checks if there are any available/invalid entries to keep the set's information. If there are no such invalid entries and if the set is less saturated than one of the sets already in the heap, replacement will take place to make room for this less-saturated one.

On the other hand, when an uncoupled taker set needs to evict a block, it first sends a coupling request to the HW heap. The heap returns the index information of the least saturated giver set for coupling, and the association table records the two sets' indices in each other's associated table entry. If there are no available giver sets in the heap, the taker just evicts the victim block off-chip.

4.6 Spilling and Receiving Control

Unlike SBC that allows a taker set to continuously evict blocks to its coupled set, our STEM LLC design imposes some restrictions on the spilling and receiving processes for any pair of coupled sets. This is because a giver set can be overwhelmed if spilling from the taker set is excessive. However, whether or not a giver set is overwhelmed can be easily detected by checking the MSB of the spatial saturating counter of its corresponding shadow set. If a previously 0-valued MSB of a spatial saturating counter turns 1, it suggests that either the set might have been overwhelmed by another set's excessive spilling or it has changed its role from a giver set to a taker set. The set-level capacity monitor returns such information to the cache controller to form a feedback loop as depicted in Figure 4. With the feedback loop, only when a set has a 0-valued MSB in the corresponding spatial saturating counter can it receive victim blocks from its coupled taker set.

While the spilling process is straightforward and similar to the SBC scheme, because only a coupled taker set can spill victim blocks to the corresponding giver set, the receiving process is significantly different. In the SBC proposal [4], it is clearly stated that receiving (using MRU insertion, namely the LRU replacement policy) is not dependent on the giver set's saturating level as long as the two sets are coupled. Such a receiving mechanism of SBC can severely pollute the giver set's space, because the taker set can excessively spill victim blocks to the giver set without looking at the actual utility of doing so. In the STEM LLC design, we set a receiving constraint so that the giver set cannot receive a foreign block unless its saturating value indicates that the set is still unsaturated even with receiving. In other words, whether or not a cooperative set is still able to contribute its capacity to the taker set can be detected by its spatial saturating counter. Furthermore, how a foreign block is inserted into the cooperative set is decided by what the cooperative set's temporal saturating counter indicates.

4.7 Decoupling Two Sets

The disassociation between two coupled sets is triggered by the event that the (former) giver set has evicted all cooperatively-cached blocks, followed by the action of re-setting the two sets' association table entries to their own original indices respectively [4]. In contrast to the SBC scheme that does not put any constraints on the spilling

Table 1. **Major Configuration Parameters**

Core	Alpha ISA, 5-Stage Pipeline 8-Wide Dispatch/Retirement 256/256 Int/Fp Registers 64/64-Entry Inst/Data TLBs 6-Int ALU, 2-Int Mul/Div, 4-Fp ALU, 2-Fp Mul/Div 64-Entry IFQ, 64-Entry LSQ, 192-Entry ROB
L1I/D	2-Way, 32KB, 64B/Line, 1/2-Cycle I/D Lat 8/16 I/D MSHRs, 8-Entry Write Buffer physically tagged and indexed (M5's built-in setting)
L2	16-Way, 2MB, 64B/Line, 6/8-Cycle Tag/Data Store Lat 64 MSHRs, 32-Entry Write Buffer physically tagged and indexed (M5's built-in setting)
Bus	16B/Cycle, 2:1 Speed Ratio, 1-Cycle Arbitration
Mem	300-Cycle Lat

and receiving processes, the decoupling process of STEM will be much faster because the taker (or giver) set will not spill (or receive) blocks after a role change of either of them is detected, which can greatly accelerate the decoupling process.

5 Experiments & Evaluation

To evaluate our STEM LLC design, in this section, we present the experimental setup, results analysis, sensitivity study and cost analysis.

5.1 Experimental Setup

We use the cycle-accurate M5 simulator [11] as our architectural simulator with the configuration listed in Table 1. The simulated processor is an Alpha21264-like [12] out-of-order core with a 5-stage pipeline. For the memory hierarchy, we model two levels of on-chip caches. The L1 instruction and data caches adopt the conventional set-associative configuration and LRU replacement policy, and we assume a coupled tag-data store organization. For the L2 cache, we model decoupled tag and data stores, and adopt the same latency parameters as those presented in [4]. Specifically, if an access to an uncoupled or coupled giver set turns out to be a miss, the latency of a tag-store access is assumed to be 6 cycles; if an access to a set is a hit, the total latency of one tag-store access and one data-store access is assumed to be 14 cycles. For SBC and STEM, if an access to a coupled taker set is a miss, and the requested block is not found in its cooperative set either, then the total latency of two consecutive tag-store accesses is 12 cycles; otherwise a second hit will cost 20 cycles in all because it involves two tag-store accesses as well as an additional data-store reference. We evaluate and compare LRU, DIP, PeLIFO, V-Way, SBC and our proposed STEM, among which both SBC and STEM may involve a second access to the cooperative set.

We select 15 benchmarks from the SPEC CPU 2000 & 2006 suites. In general, we assume that all applications can

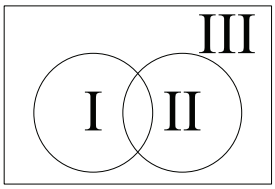


Figure 6. **Workload Classification**

be categorized into three classes according to the features of their spatial and temporal capacity demands (at the LLC set level), as shown in Figure 6. Class I includes the applications that exhibit set-level non-uniformity of capacity demands, whose performance is improvable by spatial schemes such as V-Way and SBC when the LLC capacity is in a certain range (e.g., *ammp*'s LLC performance can be improved over LRU by SBC in the associativity range [4,10], as shown in Figure 3(b)). Class II covers the programs that show poor temporal locality, so their performance is promotable by an advanced temporal scheme like DIP or PeLIFO within a certain LLC capacity range (e.g., *art*'s LLC performance can be promoted by DIP when the LLC capacity is no greater than 1MB, as demonstrated in [1]). Class III consists of such applications that show uniform set-level capacity demands as well as good temporal locality, which can be well taken care of by the simple LRU scheme. Table 2 presents these 15 benchmarks in terms of their classification as well as MPKI characteristics (under LRU).

Table 2. **The MPKI Characteristics of Benchmarks**

Class I	MPKI	Class II	MPKI	Class III	MPKI
ammp	2.535	*art*	16.769	*gobmk*	2.236
apsi	5.453	*cactusADM*	3.459	*gromacs*	1.099
astar	2.622	*galgel*	1.426	*soplex*	24.298
omnetpp	11.553	*mcf*	59.993	*twolf*	3.793
xalancbmk	14.789	*sphinx3*	10.969	*vpr*	3.306

The selected benchmarks are fast-forwarded and cache-warmed with 10 and 2 billion instructions respectively, followed by an execution of 3 billion instructions with the detailed architectural features listed in Table 1. In the evaluation, we use three performance metrics, namely, MPKI (misses per 1K instructions), AMAT (average memory access time) and CPI (cycles per instruction), to compare our STEM design against other state-of-the-art schemes in various aspects. All results are normalized to those of LRU.

5.2 Performance Analysis

Figure 7 shows the performance comparison between STEM and state-of-the-art spatial and temporal LLC management schemes, in terms of their MPKI results. For the benchmarks in Class I, as a result of STEM's capability of spatial resource management, STEM is noticeably better than the existing temporal schemes DIP and PeLIFO. Specifically, STEM outperforms the two temporal schemes by at least 12.9%, 8.1%, 53.4% and 9.7% for *ammp*, *apsi*,

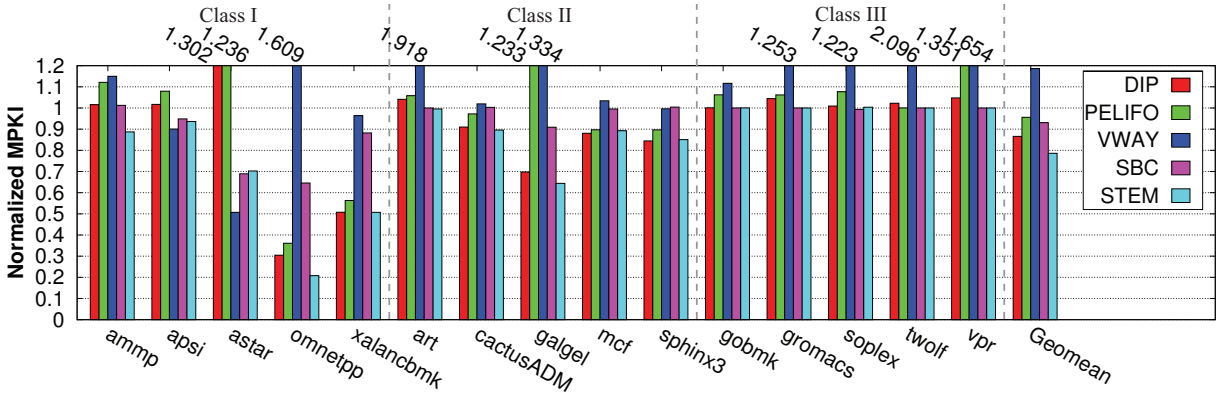


Figure 7. **Normalized MPKI**

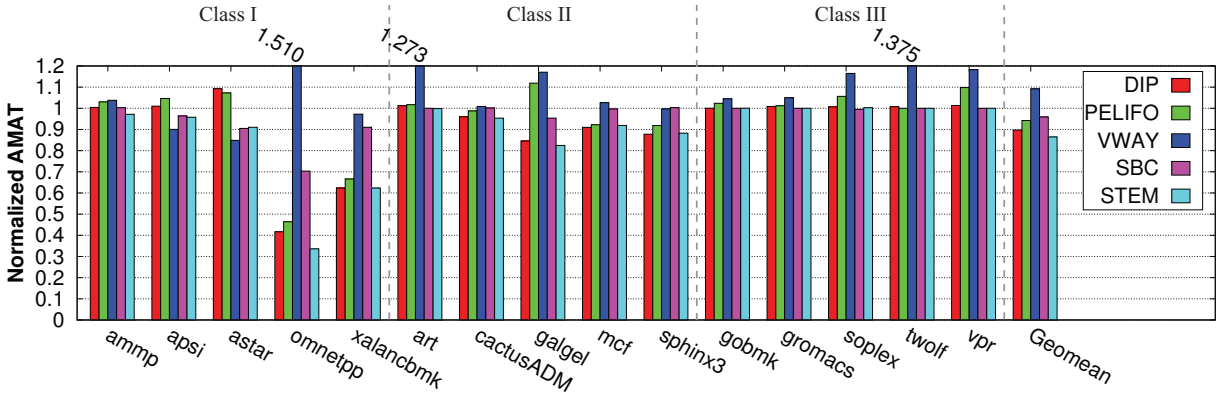


Figure 8. **Normalized AMAT**

astar and *omnetpp* respectively. Interestingly, we find that temporal schemes can degrade the MPKI performance of *astar* significantly. That is because *astar* shows obvious set-level non-uniform capacity demands and, more importantly, good temporal locality for most LLC working sets. Since temporal schemes DIP and PeLIFO both dedicate several groups of sample sets to the policy comparison, e.g., BIP versus LRU in the DIP scheme, and the policy that incurs less (in DIP) or the least (in PeLIFO) misses will be imposed upon other non-sample sets. However, due to the set-level non-uniform features, *astar*'s LLC working sets are quite different from each other, and the winning policy of the sample sets is not (necessarily) suitable for the non-sample LLC sets most of which have good temporal locality. That is why DIP and PeLIFO make inappropriate application/LLC-level replacement decisions for *astar*, e.g., BIP is the winning policy and adopted for the non-sample LLC sets in DIP. Unlike DIP and PeLIFO, STEM is able to decide on better replacement policies for individual sets based on their set-level temporal demands for certain benchmarks like *astar*.

For the five schemes in Class II, we obtain the expected better performance of temporal LLC management schemes than that of the spatial ones, because existing spatial schemes are not able to handle the cases of poor temporal locality. Since STEM also has a temporal management module, it is capable of dueling between LRU and DIP under this circumstance, but at the LLC set-level rather than at the application/LLC-level as in DIP and PeLIFO. STEM performs as well as DIP for the benchmarks of Class II. The reason why none of the schemes improves over LRU for *art* is because *art* is improvable by advanced temporal schemes only when its capacity is no greater than 1MB, as evaluated in [1], but the standard LLC capacity configured here is 2MB. With regard to the benchmarks in Class III, for which LRU is sufficient, we find that STEM performs as well as LRU and SBC that perform the best.

In Figure 7, we can also infer that the HW metric used by STEM to measure set-level capacity demands is better than those used by SBC and DIP. Of the 15 benchmarks, we see that V-Way underperforms LRU in 7 out of them, while STEM either outperforms or performs no worse than LRU. On the other hand, for the benchmarks in Class I, where spatial schemes have opportunities to significantly improve over LRU, STEM outperforms SBC, with the exception of astar for which it slightly underperforms by 0.3%. This comparison reveals that the HW metric in STEM, which

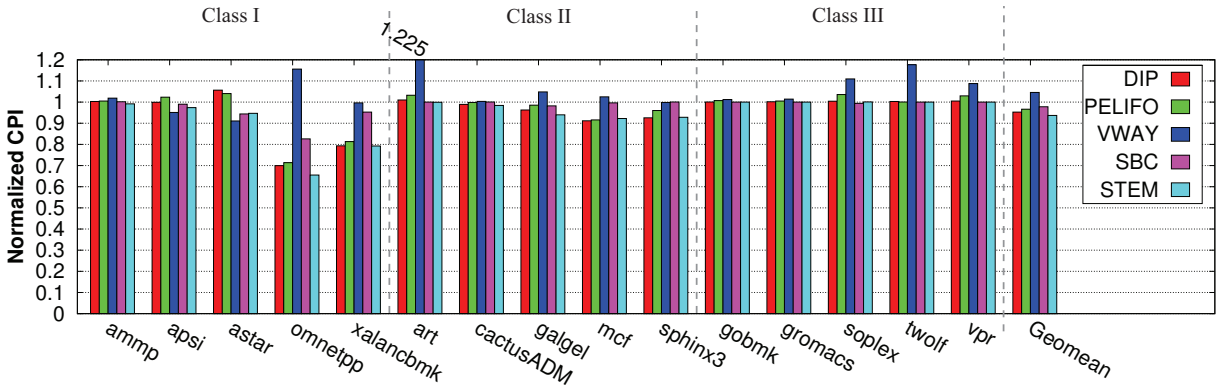


Figure 9. **Normalized CPI**

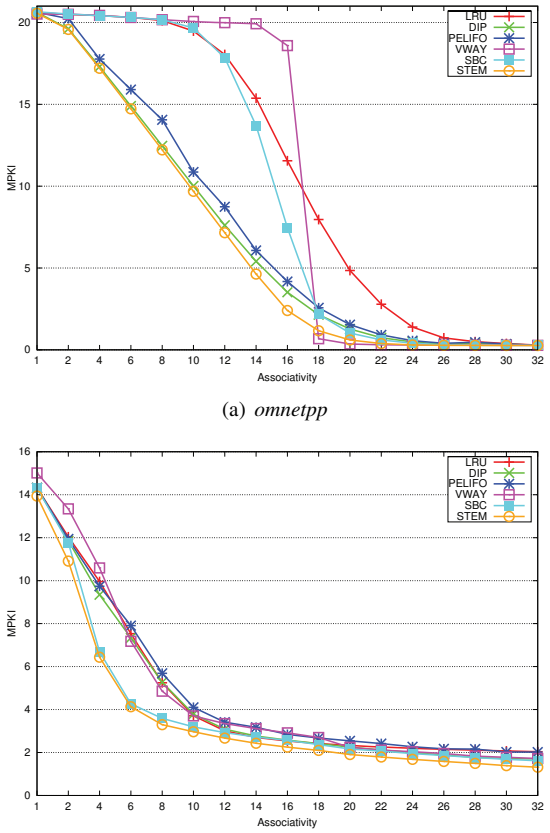
utilizes the virtual capacity of shadow tags to directly measure the benefit of extending an LLC set's capacity, is more accurate than the (implicit) metric of "access count" of V-Way as well as the "saturation level" of SBC in estimating the capacity demands of individual LLC sets.

Because both SBC and STEM can involve a second access to a cooperative LLC set, MPKI is not a direct metric for comparing the throughput of different LLC management schemes, but it sheds light on the implication of MPKI reduction on throughput. Figure 8 and Figure 9 show the AMAT and CPI results of the schemes with the timing parameters of Section 5.1 and Table 1 incorporated into the simulation. We find that the comparison results in terms of AMAT and CPI are consistent with that of MPKI in Figure 7. Specifically, the STEM LLC design can improve the AMAT measure of LRU by 13.5% and the CPI metric by 6.3%, while DIP, PeLIFO, V-Way and SBC improve the two throughput metrics by (10.3%, 4.7%), (5.8%, 3.4%), (-9.2%, -4.6%) and (4.1%, 2.2%) respectively.

All in all, benchmarks in Class I and Class II together highlight the adaptive capabilities of STEM. Specifically, STEM has generally noticeable performance advantage over existing temporal schemes for benchmark Class I, and significantly outperforms existing spatial schemes for benchmark Class II. If a benchmark belongs to both Class I and Class II, STEM can outperform both temporal and spatial schemes simultaneously, which is consistent with the extensional example shown in Figure 2. In addition, STEM is capable of deciding different replacement polices for individual LLC sets and overcoming the pathological cases that expose the weakness of advanced application/LLC-level temporal schemes; and STEM's set-level spatial capacity demand monitors that take advantage of the virtual capacity of shadow tags are shown to be more accurate than those of existing spatial schemes such as V-Way and SBC.

5.3 Sensitivity Study

We use the *omnetpp* and *ammp* benchmarks illustrated in Section 3 as examples for our sensitivity study. From



(a) *omnetpp*

(b) *ammp*

Figure 10. **Sensitivity Study**

Figure 10(a), we find that in the small associativity range of [1, 6] STEM performs as well as DIP that is the best out of all existing schemes under this condition, with noticeable performance benefit over the spatial schemes such as V-Way and SBC. In the moderate associativity range of [6, 16], STEM is able to outperform all existing LLC management schemes by combining the strengths of both spatial and temporal management. In the high associativity range of [18, 24], STEM is still be better than others except that it is

Table 3. **Hardware Analysis with the Configurations in Table 1**

address length	44-bit effective physical address in a Alpha21264 processor simulated by M5
# (LLC sets)	2048
association table	2048 entries with 11 bits each
set associativity	16
cache line size	64 bytes
tag field length	27 bits
m (the length of a shadow tag entry)	10 bits with the hash function defined in [13]
CC, V, D bits	1 bit each
replacement rank field	4 bits
k (the length of a saturating counter)	4 bits
n (2^n is the ratio that multiplies the number of hits on an LLC set in spatial measurement)	3 bits

slightly worse than V-WAY.

As illustrated in Figure 10, for *ammp*, throughout the entire experimented associativity range of [1, 32], STEM outperforms or performs no worse than the existing LLC management schemes, but with significant advantages over DIP, PeLIFO and V-Way in the associativity range of [2, 10].

From the two cases, we find that STEM is able to adapt its management strategy to both spatial and temporal capacity demands of workloads, suggesting that STEM may bridge the philosophical gap between existing spatial and temporal LLC management schemes.

5.4 Overheads Analysis

The set-level capacity demand monitor (SCDM) and association table account for the vast majority of the hardware overhead in the STEM design. Table 3 lists the length of each storage field in the STEM L2 cache. The overall storage overhead of both monitor store and association table of the LLC controller is 3.1% compared to LRU by estimation.

6 Related Work

There have been extensive studies targeted at improving the LLC performance and mitigating the memory bottleneck. As discussed in Section 2, these existing LLC management approaches can be generally categorized into either temporal or spatial dimension. The temporal schemes essentially refer to replacement policies that manage the blocks' lifetime in individual LLC sets, while the spatial mechanisms attempt to adjust space allocation among different LLC sets. In the following, several representative state-of-the-art schemes in either of the dimensions will be briefly discussed.

6.1 Temporal LLC Management

Dynamic Insertion Policy (DIP): a previous study [1] has shown that the LRU replacement policy can leave most blocks "dead"/unused between their insertion and eviction in an LLC set when the set's capacity is insufficient. In the DIP design, two small sample groups of LLC sets are dedicated to LRU and BIP modules respectively to estimate and compare the performance of assigning incoming blocks with the longest or shortest lifetime, and the winning policy is adopted to other non-sample sets to either exploit temporal locality or prevent thrashing.

Probabilistic Escape LIFO (PeLIFO): Another recent research [2] takes advantage of a fill-stack to rank the cached block for a set according to the Last-In-First-Out order. The PeLIFO policy is unique in that it typically does not evict the block with the lowest rank in the fill-stack on replacement, but instead learns the most preferred eviction positions close to the top of the fill-stack and evicts the one from the position that can lead to the best performance.

Cache Bursts: Based on the observations that cache blocks inserted by the same memory instruction have similar reuse patterns and that it is more accurate to trigger data-block prediction only when a block becomes non-MRU, Liu et al. [5] have devised the *Cache Bursts* mechanism to more accurately identify a "dead" block based on a history table. Once identified, the dead blocks can be replaced much earlier than through LRU, proactively making room for incoming blocks.

The proposals above focus on designing alternative replacement policies to better temporally manage the capacity of individual overutilized sets. However, they cannot utilize the available space existing among underutilized sets, rendering it less effective when set-level space utilization is unbalanced. Our STEM LLC management scheme is capable of not only enabling underutilized sets to share space with overutilized ones but also optimizing a set's temporal behavior of utilizing the capacity it has, thus significantly outperforming state-of-the-art temporal schemes when workloads expose prominent non-uniform set-level capacity demands but being at least no worse in other cases.

6.2 Set-Level Spatial LLC Management

Variable-Way Set Associativity (V-Way): it is observed in [3, 6, 14] that there exists a non-uniform distribution of accesses to different LLC sets under many workloads, and the skewed associativity [14] and the prime-based set indexing [6] were the early work that diffuses accesses to different sets in a more balanced way. A later scheme called Variable-Way LLC [3] is proposed to dynamically adjust the number of data lines assigned to each tag set depending on the set's access pattern. Since the V-Way cache has twice (or multiple times) as many tag entries as data lines, the association between a tag entry and a data line needs to be dynamically established by using a pair of front and backward pointers. In addition, tag entires and data lines are replaced by using LRU and a global frequency-based replacement policy respectively. These features allows V-WAY to be adaptive to the non-uniform set-access distribution and better spatially manage the LLC capacity.

Set Balancing Caches (SBC): It is observed in [4] that the difference between the miss and hit counts of a set, which is defined as the set's saturation level, varies from set to set in LLC. A higher saturation level of a set is assumed to indicate a greater capacity demands than the set's current space. The SBC scheme is developed to balance the non-uniform set-demand by pairing a saturated set exhibiting a high saturation level with another set having a low saturation level, which enables the saturated set to place victim blocks in the other set.

Our STEM LLC management scheme differs from the aforementioned spatial LLC approaches in at least two aspects. First, STEM treats the signature-based shadow sets as extra virtual capacity and thus directly measures the performance benefit of extending a set's capacity, which enables it to be very accurate in calibrating the set-level capacity demands. On the contrary, as explained in Section 3.1 and Section 5.2, the "access count" of V-Way and "saturation level" of SBC are both indirect approximation metrics and thus less effective in estimating a set's capacity needs. Second, STEM is able to decide the best temporal behavior of utilizing the capacity (either local or cooperative) accessible to a set, but neither V-Way nor SBC has such a salient feature, which leads to a much better performance of STEM over the two.

6.3 Page-Level Spatial Management

Page Coloring: page coloring is an OS-based approach for cache management by manipulating an overlapped section (namely page color) between physical page index and LLC set index when mapping a block from a physical page to a cache set. A recent proposal [7] called the Run-time Operating system Cache-filtering Service (ROCS) designates a small LLC cache region that a physical page can fit in as a pollute buffer. Pages exhibiting high miss rates are re-colored and remapped to the pollute buffer, preventing others with high hit rates from pollution. This kind of software-based page coloring schemes are very flexible in implementation, but the expensive re-coloring requires flushing off a page's cached blocks and migrating them from one memory frame to another in main memory. Therefore, this software approach is only applicable to workloads with relatively long stable phases that can offset the high re-coloring cost.

7 Conclusions

This paper proposes a novel LLC design, which is called STEM (*SpatioTEmporally Managed*) LLC, to dynamically identify both spatial and temporal dimensions of capacity demands at the set level, couple two sets with complementary spatial resource needs for inter-set capacity sharing and decide on the best replacement policies for individual LLC sets. Our executing-driven simulation shows that the STEM LLC design can improve the performance metrics of MPKI (misses per 1k instruction), AMAT (average memory access time) and CPI (cycle per instruction) over LRU by 21.4%, 13.5% and 6.3% respectively, better than the improvements obtained by the state-of-the-art DIP, PeLIFO, V-Way and SBC LLC management schemes, at a manageable HW storage cost of only 3.1%.

Acknowledgement

We would like to thank UNL RCF (Research Computer Facility) staff for their help with the experiment setup, and thank Lei Tian for his inputs to improve the paper. We also greatly appreciate the constructive comments and suggestions from the anonymous reviewers. This work was supported in part by NSF under Grants CCF-0621526, CCF-0937993, IIS-0916859 and CNS-1016609.

References

[1] M. K. Qureshi, A. Jaleel, Y. N. Patt, S. C. Steely, and J. Emer, "Adaptive Insertion Policies for High Performance Caching," in *Proceedings of the 34th Annual International Symposium on Computer architecture*, pp. 381–391, 2007.

[2] M. Chaudhuri, "Pseudo-LIFO: the Foundation of a New Family of Replacement Policies for Last-Level Caches," in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 401–412, 2009.

[3] M. K. Qureshi, D. Thompson, and Y. N. Patt, "The V-Way Cache: Demand Based Associativity via Global Replacement," in *Proceedings of the 32nd Annual International Symposium on Computer Architecture*, pp. 544–555, 2005.

[4] D. Rolán, B. B. Fraguela, and R. Doallo, "Adaptive Line Placement with the Set Balancing Cache," in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 529–540, 2009.

[5] H. Liu, M. Ferdman, J. Huh, and D. Burger, "Cache Bursts: A New Approach for Eliminating Dead Blocks and Increasing Cache Efficiency," in *Proceedings of the 41st annual IEEE/ACM International Symposium on Microarchitecture*, pp. 222–233, 2008.

[6] M. Kharbutli, K. Irwin, Y. Solihin, and J. Lee, "Using Prime Numbers for Cache Indexing to Eliminate Conflict Misses," in *Proceedings of the 10th International Symposium on High Performance Computer Architecture*, pp. 288–299, 2004.

[7] L. Soares, D. Tam, and M. Stumm, "Reducing the Harmful Effects of Last-Level Cache Polluters with an OS-Level Software-Only Pollute Buffer," in *Proceedings of the 41st Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 258–269, 2008.

[8] D. Zhan, H. Jiang, and S. C. Seth, "Exploiting Set-Level Non-Uniformity of Capacity Demand to Enhance CMP Cooperative Caching," in *Proceedings of the 24th annual IEEE International Parallel & Distributed Processing Symposium*, pp. 222–233, 2010.

[9] L. A. Belady, "A Study of Replacement Algorithms for a Virtual-Storage Computer," *IBM Systems Journal*, vol. 5, no. 2, pp. 78–101, 1966.

[10] T. R. Puzak, *Analysis of Cache Replacement Algorithms*. PhD thesis, 1985.

[11] N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt, "The M5 Simulator: Modeling Networked Systems," *IEEE Micro*, vol. 26, no. 4, pp. 52–60, 2006.

[12] R. E. Kessler, "The Alpha 21264 Microprocessor," *IEEE Micro*, vol. 19, no. 2, pp. 24–36, 1999.

[13] M. V. Ramakrishna, E. Fu, and E. Bahcekapili, "Efficient Hardware Hashing Functions for High Performance Computers," *IEEE Transactions on Computers*, vol. 46, no. 12, pp. 1378–1381, 1997.

[14] A. Seznec, "A Case for Two-Way Skewed-Associative Caches," *SIGARCH Computter Architecture News*, vol. 21, no. 2, pp. 169–178, 1993.