IEICE TRANS. INF. & SYST., VOL.E105–D, NO.7 JULY 2022

LETTER

A Two-Level Cache Aware Adaptive Data Replication Mechanism for Shared LLC

Qianqian WU[†a)], ***Student Member and*** **Zhenzhou JI**[†b)], ***Nonmember***

SUMMARY The shared last level cache (SLLC) in tile chip multiprocessors (TCMP) provides a low off-chip miss rate, but it causes a long on-chip access latency. In the two-level cache hierarchy, data replication stores replicas of L1 victims in the local LLC (L2 cache) to obtain a short local LLC access latency on the next accesses. Many data replication mechanisms have been proposed, but they do not consider both L1 victim reuse behaviors and LLC replica reception capability. They either produce many useless replicas or increase LLC pressure, which limits the improvement of system performance. In this paper, we propose a two-level cache aware adaptive data replication mechanism (TCDR), which controls replication based on both L1 victim reuse behaviors prediction and LLC replica reception capability monitoring. TCDR not only increases the accuracy of L1 replica selection, but also avoids the pressure of replication on LLC. The results show that TCDR improves the system performance with reasonable hardware overhead.

key words: *tiled chip multiprocessors (TCMP), shared last level cache (SLLC), replication, L1 victim reuse behaviors, LLC replica reception capability*

1. Introduction

Tiled chip multiprocessor (TCMP), which contains a series of identical tiles connected over an unordered point-to-point on-chip network, is becoming a more and more practical processor design due to its scalability of multicore and manycore* [1]. In TCMP, the shared last level cache (SLLC) is the mainstream design, that is always evenly divided into slices equal to the number of tiles. Due to the large shared cache capacity, the off-chip miss rate of LLC is low, which is positive to the improvement of system performance. However, if the requested cache line is located in a remote tile, SLLC will cause a long on-chip access latency, which hurts the system performance.

Victim Replication (VR) [2] is first proposed to address the long on-chip access latency problem of SLLC, which replicates L1 victims to local LLC slice in the same tile. However, VR does not consider not only the L1 victim reuse behaviors, but also the LLC replica reception capability. Although many improved data replication mechanisms [3]–[5] are proposed after VR, the L1 victim reuse behaviors and the LLC replica reception capability are not considered at the same time. Previous replication schemes will produce a large number of useless replicas or increase LLC pressure, and VR has both disadvantages. Thus, in all previous mechanisms, the improvement of system performance is limited.

In this paper, a two-level cache aware adaptive data replication mechanism TCDR is proposed. At the L1 cache level, TCDR predicts the victim reuse behaviors and selects victims with high reuse locality and/or short reuse distance as replicas to achieve high replication accuracy. At the LLC level, TCDR monitors the replica reception capability to determine whether replicas are inserted into MRU or LRU positions to avoid the pressure of replication on LLC. In addition, when the level of replica reception capability changes, LLC will feed it back to L1 to update the replica selection criteria. The results show that TCDR is superior to the baseline data replication mechanism in performance.

2. Related Work

(1) Data replication. VR [2] replicates all L1 victims to local LLC slices without a replica selection process based on the L1 victim reuse behaviors. Besides, VR inserts all replicas to the MRU position ignoring LLC replica reception capability. Although Adaptive Selective Replication (ASR) [3] makes a replica selection at L1 level and weighs the benefits and losses at LLC level, ASR only selects shared read-only victims for replication at the L1 level and blindly replicate all shared read-only victims as replicas without considering the L1 victim reuse behaviors. Several other works also explore data replication in SLLC [4], [5]. However, these schemes focus on either the L1 replica selection strategy or the LLC replica reception capability. In contrast, TCDR controls data replication based on both L1 victim reuse behaviors prediction and LLC replica reception capability monitoring. TCDR makes a good trade-off between the on-chip access latency and capacity of LLC. Thus, TCDR can gain a more significant improvement in performance than all previous schemes.

(2) Reuse locality, reuse distance prediction and set dueling. Reuse locality [6] has been applied to manage SLLC. It means that the cache lines that have been used in SLLC may no longer be used, but the reused cache lines tend to be reused many times in the near future. So, cache accesses

Manuscript received January 8, 2022.
Manuscript revised February 28, 2022.
Manuscript publicized March 25, 2022.

[†]The authors are with the School of Computer Science and Technology, Harbin Institute of Technology, No.92 Xidazhi Street, Nangang District, Harbin City, Heilongjiang Province, China.
a) E-mail: wqqhit@163.com
b) E-mail: jizhenzhou@hit.edu.cn (Corresponding author)

DOI: 10.1587/transinf.2022EDL8002

*In this paper, we assume TCMP has a two-level cache hierarchy, so LLC corresponds to L2 cache and data replication acts between L1 and L2 cache. In a three-level cache hierarchy, LLC corresponds to L3 cache and data replication acts between L2 and L3 cache.

in SLLC show reuse locality rather than locality. Signature-based hit predictor (SHiP) [7] uses reuse distance prediction to improve SLLC replacement. SHiP adopts a signature history counter table (SHCT) of saturation counters to learn the reuse behavior of a signature such as program counter (PC). Besides, Dynamic Insertion Policy (DIP) uses set dueling [8] in cache insertion policy. Set dueling leverages the fact that LLC has a large number of sets and cache performance can be estimated by sampling a few sets. Specifically, a few dedicated sets are selected for two competitive strategies, and the winning strategy is applied to the rest follower sets. Inspired by the above works [6]–[8], TCDR introduces reuse locality and reuse distance prediction for L1 replica selection to improve accuracy and uses set dueling for LLC reception capability monitoring to compete between LRU and MRU insertion position.

3. Design of TCDR

3.1 Organization

Figure 1 shows the organization structure of TCDR, which is deployed to each tile of TCMP. On one hand, at the L1 cache level, we add 3-part structures for predicting L1 victim reuse behaviors, including reuse locality and reuse distance. Firstly, each entry in L1 cache is extended with a PC part. Secondly, we add instruction and data victim tables (VTT_I and VTT_D) to store tag and PC of victims evicted from L1 instruction and data cache respectively. PC is used for victim reuse distance prediction. Each entry in VTT is extended with a reuse (Re) bit for victim reuse locality prediction. "1" indicates victim has been reused and has a high reuse locality. VTT adopts set-associative structure and LRU replacement policy. Thirdly, we introduce a one-dimensional PC history counter table (PHCT) similar to that in SHiP for victim reuse distance prediction. Each entry in PHCT is indexed by a hash of PC and contains a saturation counter (SC) to record the history information for reuse distance prediction. SC not equal to "0" indicates victim has a short reuse distance. On the other hand, at the LLC level, since we use set dueling to monitor the LLC replica reception capability, we introduce a policy selector (PSEL) which has been used in DIP. PSEL is a saturation counter that tracks which of the two competitive replica insertion policies (insert to the MRU or LRU position) causes fewer misses. The most significant bit (MSB) of PSEL indicates the level of LLC replica reception capability, "0" stands for "strong" and "1" for "weak".

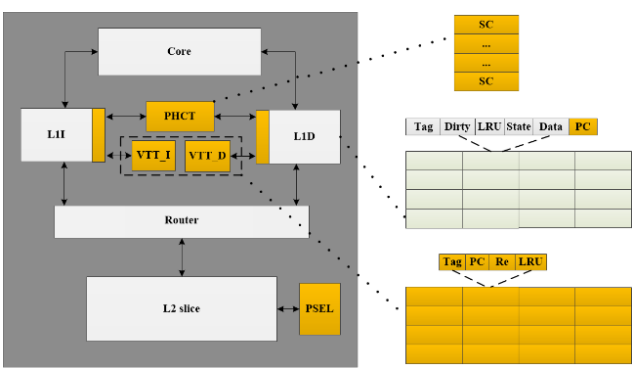

Fig. 1 The organization structure of TCDR

3.2 Algorithms

(1) L1 replica selection based on victim reuse behaviors prediction. In order to use the prediction of L1 victim reuse behaviors to guide the selection of replicas, the victim reuse behaviors of reuse locality and reuse distance should be learned during the program execution (line1~line5 in algorithm1). When an L1 access request misses or L1 evicts a victim, VTT and PHCT need to be accessed and updated. In the former situation, if VTT hits, its Re is set to "1". Besides, the SC in PHCT index by a hash of PC is increased by 1. In the latter situation, if an entry is evicted from VTT but the Re is equal to "0", the SC in PHCT index by a hash of PC is decreased by 1. Then, the learning result of L1 victim reuse behaviors will be used to L1 replica selection (line6~line11 in algorithm1). TCDR defines two replica candidate sets (Set_1: high reuse locality and Set_2: short reuse distance). When L1 evicts a victim, it selects replicas in $Set_1 \cup Set_2$ or $Set_1 \cap Set_2$ according to the level of LLC replica reception capability (The level is monitored and fed back to L1 in the following algorithm2). If VTT hits, the L1 victim will be replicated to the local LLC in two cases. First, the level is "strong", and the Re is equal to "1" or the SC in PHCT index by a hash of PC is not equal to "0"; Second, the level is "weak", and the Re is equal to "1" and the SC in PHCT index by a hash of PC is not equal to "0". If VTT misses, the L1 victim will be replicated to the local LLC when the level is "strong", and the SC in PHCT index by a hash of PC is not equal to "0".

(2) LLC replica dynamic insertion based on replica reception capability monitoring. First of all, the LLC replica

Algorithm1: L1 replica selection based on victim reuse behaviors prediction

When an L1 access request misses:

1: if the VTT access request hits
2: **Re** ← 1;
3: ++**PHCT**[hash(**PC**)];

When L1 evicts a victim:

4: if victim(**VTT**).**Re** == 0
5: –**PHCT**[hash(**PC**)];
6: if the VTT access request hits
7: if [**strong** & (**Re** == 1 || **PHCT**[hash(**PC**)] != 0)] ||
*[**weak** & (**Re** == 1 & **PHCT**[hash(**PC**)] != 0)]*
8: **Replicate** L1 victim to LLC;
9: else
10: if [**strong** & (**PHCT**[hash(**PC**)] != 0)]
11: **Replicate** L1 victim to LLC;

Algorithm2: LLC replica dynamic insertion based on replica reception capability monitoring

When an LLC access request misses:

1: if set ∈ **SDM-M**
2: **PSEL** += 16;(home) / ***PSEL*** *+= 1;(local)*
3: if set ∈ **SDM-L**
4: **PSEL** −= 16;(home) / ***PSEL*** *-= 1;(local)*
5: if **MSB(PSEL)** changes from **1** to **0**
6: level ← **strong**;
7: Feedback **"strong"** level to L1;
8: if **MSB(PSEL)** changes from **0** to **1**
9: level ← **weak**;
10: Feedback **"weak"** level to L1;

When a new replica (NR) from L1 arrives:

11: if set(NR) ∈ **SDM-M**
12: Insert NR to **MRU** position;
13: else if set(NR) ∈ **SDM-L**
14: Insert NR to **LRU** position;
15: else // Follower sets
16: if level == **strong**
17: Insert NR to **MRU** position;
18: else
19: Insert NR to **LRU** position;

reception capability needs to be monitored using set dueling to compete LRU and MRU insertion position (line1∼line10 in algorithm2). We assume that each LLC slice contains 256 sets. Following DIP, we select 16 dedicated sets to insert a new replica to the MRU position to form SDM-M (Set Dueling Monitor–MRU), and select 16 non-overlapping dedicated sets to insert a new replica to the LRU position to form SDM-L (LRU). The rest follower sets apply the winning policy between SDM-M and SDM-L. The dedicated sets can be selected based on the comparison results of the higher four bits ([7:4]) and the lower four bits ([3:0]) of the set index. If the SetIndex[7:4] is equal to SetIndex[3:0], the dedicated set belongs to SDM-M, and if the SetIndex[7:4] is equal to the complement of SetIndex[3:0], the dedicated set belongs to SDM-L. The home LLC access miss will cause an off-chip memory access, requiring an access latency of 300 cycles; while the local LLC access miss requires forwarding the request to the remote home LLC slice, requiring an average access latency of 18 cycles. The cost of home misses is about 16 times that of local misses. Therefore, when an LLC access request misses, a home miss in SDM-M increases PSEL by 16 and a local miss increases PSEL by 1; A home miss in SDM-L decreases PSEL by 16 and a local miss decreases PSEL by 1. What's more, the replica reception capability is indicated by the MSB of PSEL. If the MSB changes from "1" to "0", it means that the replica reception capability level becomes "strong". If the MSB changes from "0" to "1", the level becomes "weak". The level changes are captured and fed back to L1. Then, the level of replica reception capability is used to guide L1 replica selection (line6∼line11 in algorithm1) and LLC replica dynamic insertion (line11∼line19 in algorithm2). When a new replica (NR) from L1 arrives, it will be inserted to the MRU or LRU position dynamically. If the set of NR belongs to SDM-M, insert NR to the MRU position; if it belongs to SDM-L, insert NR to the LRU position. When the set of NR belongs to follower sets, the "strong" level makes NR be inserted to the MRU position. Otherwise, insert NR to the LRU position.

4. Experiments

We evaluate TCDR using gem5 [9] simulator. TCDR is implemented on the basis of a 64-tiled CMP system with private L1 cache and shared LLC, and the on-chip interconnection mode adopts 8*8 mesh network. System parameters are shown in Table 1. In addition, we use workloads from PARSEC [10] and SPLASH-2 [11] as the benchmark.

4.1 Performance

We use execution time as the performance metric, and the lower the execution time, the better the performance.

(1) Comparison with the fixed reception levels

In order to understand the advantage of the proposed TCDR mechanism more clearly, we compare TCDR with two cases where the reception level is fixed to "strong" and "weak". When the level is fixed to "strong", L1 victims with high reuse locality or short reuse distance are always inserted to the MRU position of LLC. When the level is fixed to "weak", L1 victims with high reuse locality and short reuse distance are always inserted to the LRU position of LLC. Figure 2 shows the performance of TCDR and the two fixed levels. Thanks to the set dueling method, the result of the

Table 1 System parameters

Parameter	Value
Processor	64 cores, 2 GHz
L1 Cache	Split I&D, Private, 16KB, 4-way, LRU, 1 cycle
LLC (L2 Cache)	Inclusive, Shared, 128KB/tile, 8-way, LRU, 6 cycles
DRAM	300 cycles
Network	8*8 mesh, 6 cycles (router: 2 cycles, link: 4 cycles)

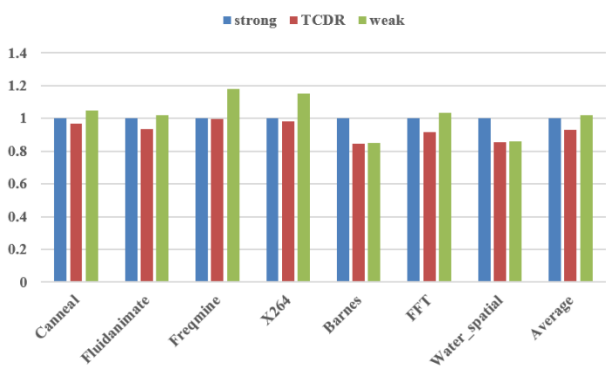

Fig. 2 Performance normalized to the fixed "strong" level

Table 2 The hardware overhead of TCDR

System	Structure	Entry Size	Entry Number	Size	Total Size	Overhead
Baseline	L1_I/D	64B	256×2	32KB	160KB	0.0%
	LLC	64B	2K	128KB		
TCDR	PHCT(SC)	3-bit	16K	6KB	10.001KB	6.3%
	L1_I/D(PC)	14-bit	256×2	0.875KB		
	VTT_I/D(Tag/PC/Re/V/LRU)	50-bit	256×2	3.125KB		
	PSEL	9-bit	1	0.001KB		

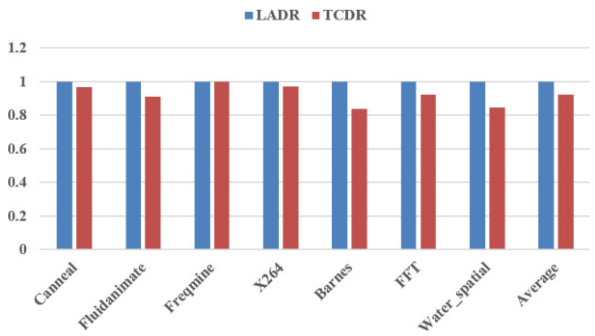

Fig. 3 Performance normalized to LADR

proposed TCDR mechanism is basically similar to the better of the fixed "strong" and "weak" levels (such as "Freqmine" and "Barnes"). TCDR can even surpass both of them for benchmarks such as "Fluidanimate" and "FFT" because the level is adaptively selected. In general, TCDR improves the performance by 7.11% and 9.87% respectively compared with the fixed "strong" and "weak" levels. This proves the advantage of the adaptive method in this paper.

(2) Comparison with LADR

We choose a closely related data replication mechanism LADR [4] as the baseline system for performance evaluation. We select LADR as the comparison target because it is a state-of-art data replication mechanism that benefits from the replication of all types of data, which is consistent with TCDR in this paper, while other replication mechanisms [3], [5] only replicate read-only data. So, taking LADR as the comparison target can better reflect that the performance improvement is due to our proposed two-level mechanism. Moreover, LADR has proved that its performance is better than VR [2] and ASR [3], so we can indirectly prove that our TCDR is superior to VR and ASR in performance.

Figure 3 shows the performance of TCDR normalized to LADR. TCDR improves the performance by 7.83% than LADR on average. Because LADR only considers the reuse times of L1 victims, and does not consider the LLC replica reception capability. This will not make rational use of the idle resources of LLC, which will affect the system performance. TCDR can avoid this disadvantage by sensing two-level caches and adaptively adjusting the replication strategy. In addition, TCDR and LADR have similar performance at the benchmark "Freqmine" (TCDR is only 0.2% better than LADR). The reason is that the level of LLC replica reception capability is "strong" in most cases, LADR will not increase LLC pressure due to neglect of LLC replica reception capability. And TCDR can sense LLC replica reception capability and adopt reasonable L1 replica selection and LLC replica insertion strategies to improve performance.

4.2 Latency Overhead

When a replacement occurs in L1 cache, the replica selection process is triggered, and serial access to VTT and PHCT is required. Besides, when a new replica needs to be inserted to LLC, PSEL will be accessed to decide the insertion position. The latency of VTT, PHCT and PSEL is 1 cycle evaluated using cacti 6.5 [12] at 32 nm technology. However, the serial access to VTT and PHCT occurs only when L1 is replaced rather than all L1 misses, and the access to PSEL occurs only when a new replica comes rather than all LLC insertion operations. Moreover, not all L1 victims will become replicas because of replica selection, which greatly reduces the number of new replicas in LLC. Therefore, the additional latency in TCDR is relatively less. In addition, by converting remote LLC accesses to local accesses, TCDR can reduce the on-chip access latency by at least 6 cycles (the nearest tile) and up to 36 cycles (the most remote tile), which will offset the additional latency overhead and improve the system performance.

4.3 Hardware Overhead

The hardware overhead of TCDR is shown in Table 2. We select 16 KB L1 instruction/data (I/D) cache and 128 KB LLC slice on the same tile as the baseline on-chip storage system. TCDR adds four additional hardware structures. Firstly, similar to SHiP, PHCT contains 16 K entries, and each entry contains a 3-bit SC, so the hardware overhead introduced in the first part is 6 KB. Secondly, both L1 instruction cache and L1 data cache have a capacity of 16 KB and the size of each cache block is 64 B, so the number of L1 cache entries is 256×2. In order to index 16 K entries in PHCT, the tag part of each entry in L1 cache is extended to store 14-bit hashed PC. Therefore, the hardware overhead introduced in the second part is 0.875 KB. Thirdly, VTT uses the same sets and ways as L1, so its number of entries is also 256×2. Each entry in VTT occupies 50 bits of storage space, including 32-bit tag, 14-bit hashed PC, 1-bit Re, 1-bit Valid (V) and 2-bit LRU, so the hardware overhead introduced in the third part is 3.125 KB. Lastly, in DIP, 32 dedicated sets use 10-bit PSEL, and 16 dedicated sets

in TCDR need 9-bit PSEL (0.001 KB). Therefore, the total hardware overhead introduced by the four parts of TCDR is 10.001 KB, accounting for 6.3% of the baseline. The experimental results show that the above settings can meet the needs of improving system performance and reasonable storage overhead.

It's worth noting that the additional hardware structures are used to store information for L1 victim reuse prediction and LLC replica reception capability monitoring, rather than store the cache data. So, the additional 10.001 KB capacity in TCDR refers to the space size of the newly added hardware structures (PHCT, PC, VTT and PSEL), not the increased cache capacity (the cache capacity is still 160 KB). In fact, in order to realize locality-aware data replication, LADR also adds a new structure (complete locality classifier) to store locality information, which introduces 48.5 KB additional hardware space. Although the hardware overheads in TCDR (10.001 KB) and LADR (48.5 KB) are not equal, the cache capacity in TCDR and LADR is still equal (still 160 KB). In addition, the latencies of additional hardware structures in TCDR and LADR are accounted respectively in our performance evaluation. Therefore, the difference of additional hardware capacity in TCDR and LADR will not affect the fairness of performance evaluation.

5. Conclusion

In order to alleviate the long on-chip access latency problem of SLLC, we propose a novel two-level cache aware adaptive data replication mechanism called TCDR. TCDR controls data replication by predicting the victim reuse behaviors at the L1 cache level and monitoring the replica reception capability at the LLC level. The results show that TCDR improves the system performance by 7.83% on average than the baseline system. Besides, the latency overhead introduced by the new structures can be offset by the reduced on-chip access latency and the hardware overhead is reasonable compared to the baseline on-chip storage system.

Acknowledgments

This work is supported by the National Key Research and Development Project (Key Technologies and Applications of Security and Trusted Industrial Control System under grant No.2020YFB2009500).

References

[1] S. Das and H.K. Kapoor, "Victim retention for reducing cache misses in tiled chip multiprocessors," Microprocess and Microsystem, vol.38, no.4, pp.263–275, 2014.
[2] M. Zhang and K. Asanovic, "Victim replication: maximizing capacity while hiding wire delay in tiled chip multiprocessors," IEEE International Symposium on Computer Architecture, pp.336–345, IEEE, 2005.
[3] B.M. Beckmann, M.R. Marty, and D.A. Wood, "ASR: adaptive selective replication for cmp caches," International Symposium on Microarchitecture, pp.443–454, IEEE, 2006.
[4] G. Kurian, S. Devadas, and O. Khan, "Locality-aware data replication in the last-level cache," International Symposium on High-Performance Computer Architecture, pp.1–12, IEEE, 2014.
[5] P.A. Tsai, N. Beckmann, and D. Sanchez, "Nexus: a new approach to replication in distributed shared caches," International Conference on Parallel Architectures and Compilation Techniques, pp.166–179, IEEE, 2017.
[6] J. Albericio, P. Ibáñez and J.M. Llabería, "Exploiting reuse locality on inclusive shared last-level caches," ACM Transactions on Architecture and Code Optimization, vol.9, no.4, pp.1–19, 2013.
[7] C.-J. Wu, A. Jaleel, W. Hasenplaugh, M. Martonosi, S.C. Steely, and J. Emer, "SHiP: Signature-based Hit Predictor for high performance caching," International Symposium on Microarchitecture, pp.430–441, IEEE, 2011.
[8] M.K. Qureshi, A. Jaleel, Y.N. Patt, S.C. Steely, and J. Emer, "Adaptive insertion policies for high performance caching," International Symposium on Computer Architecture, pp.381–391, ACM, 2007.
[9] N. Binkert, B. Beckmann, G. Black, S.K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D.R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M.D. Hill, and D.A. Wood, "The gem5 simulator," ACM SIGARCH Computer Architecture News, vol.39, no.2, pp.1–7, 2011.
[10] C. Bienia, S. Kumar, J.P. Singh, and K. Li, "The PARSEC benchmark suite: Characterization and architectural implications," International Conference on Parallel Architectures and Compilation Techniques, pp.72–81, 2008.
[11] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, and A. Gupta, "The SPLASH-2 programs: characterization and methodological considerations," ACM SIGARCH Computer Architecture News, vol.23, no.2, pp.24–36, 1995.
[12] S.J.E. Wilton and N.P. Jouppi, "CACTI: an enhanced cache access and cycle time model," IEEE Journal of Solid-State Circuits, vol.31, no.5, pp.677–688, 1996.