International Journal of Grid and Distributed Computing
Vol. 9, No. 11 (2016), pp.81-94
http://dx.doi.org/10.14257/ijgdc.2016.9.11.07

DH-LRU: Dynamic Hybrid LRU Caching Scheme for PRAM/DRAM Hybrid Main Memory

Yongjian Ren[1], Hongtianchen Xie[1], Gangyong Jia[1,2], Jilin Zhang[1,2], Yuyu Yin[1,2]
and Jian Wan[3]

[1] Department of Computer Science and Technology, Hangzhou Dianzi University
Hangzhou, 310018, China
[2] Key Laboratory of Complex Systems Modeling and Simulation, Ministry of Education
Hangzhou, 310018, China
[3] School of Information and Electronic Engineering, Zhejiang University of Science and Technology
Hangzhou, 310023
gangyong@hdu.edu.cn

Abstract

Both performance and capacity of the main memory are the key to the computer systems in current architecture. DRAM, which is the most used main memory, can't extend in capacity for its high energy consumption and repeatedly refresh. Fortunately, some new memory mediums, such as phase-change memory (PRAM), are used to replace traditional DRAM memory. These new memories have many advantages, like low energy consumption, without repeatedly refresh, high density storage, and so on. Therefore, these memories are promising. However, their low read/write performance and limited life are restricted the replacement process. In current time, hybrid memory, which consists of both PRAM and DRAM, is a good choice. In this way, the memory capacity can be extended. So, the most challenge for the hybrid memory is the performance. In this paper, we propose a dynamic hybrid LRU caching scheme (DH-LRU) for the last level cache in PRAM/DRAM hybrid main memory to improve the main memory performance. Compared with traditional cache policies, like LRU, FIFO, RANDOM, CFLRU, our DH-LRU improves performance by 4.6%. Moreover, energy consumption of write and read operation can be reduced up to 88.2%.

***Keywords**: phase change memory; hybrid memory architecture; cache policy; performance; energy.*

1. Introduction

Both performance and capacity for the main memory are important for the whole computer system. Moreover, the demand of memory for applications are more. Therefore, the demand of memory in the computer becomes more. However, currently used main memory is DRAM, which is high energy consumption and needs repeatedly refresh. So, it's restrict to extend large capacity for the DRAM [1].

Samsung Electronics tried to use Phase change memory (PRAM), which is a new memory medium, in mobile phones in 2010 [2]. Moreover, they present a 20nm 1.8V 8G PRAM at the 2012 International Solid State Circuits Symposium [3]. Phase change memory (PRAM) is proposed in current to satisfy the large main memory capacity for its low energy consumption, without repeatedly refresh, high density storage [17], non-volatile [18], and so on. PRAM is a promising storage medium for the future.

However, PRAM [4, 5] has some drawbacks compared with DRAM. Especially, its long read/write latency and limited life are restrict PRAM replace DRAM immediately. TABLE 1 shows the parameter comparison between PRAM and DRAM. Through this table, the read latency of PRAM is longer. But the worst part is the write latency. PRAM is almost 10 times than DRAM, which will make poor performance of the main memory.

Table 1. Comparison Between PRAM and DRAM

Attributes	DRAM	PRAM
Read Latency	50ns	80ns
Write Latency	30~80ns	200~800ns
Average Energy	~0.1nJ/b	~0.3nJ/b
Idle Power	~1.3W/GB	~0.5W/GB
Endurance	∞	$10^8$for write

In order to take advantage of PRAM, some hybrid main memory architectures have proposed to improve memory performance. Figure 1 has shown two architectures of hybrid main memory.

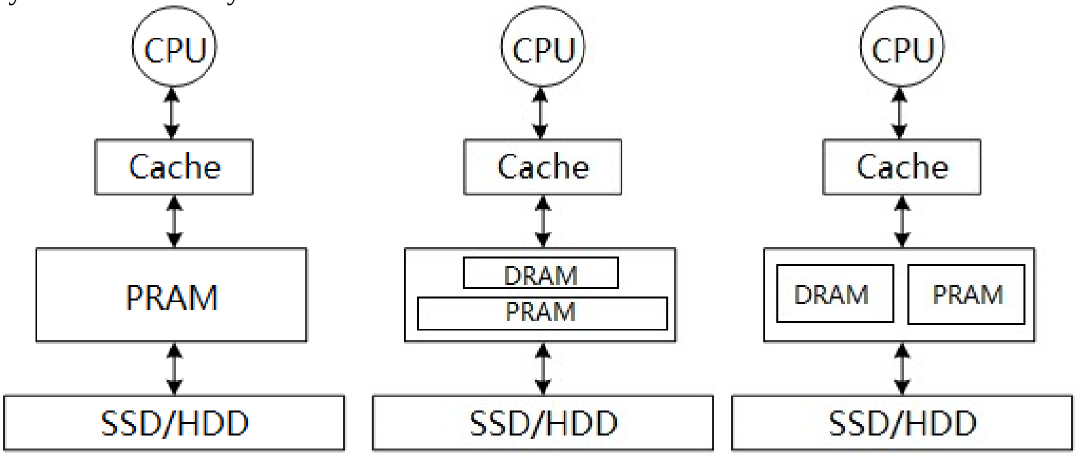

Figure 1. Three architectures of Hybrid PDRAM

The left figure in Figure 1 shows the first type hybrid main memory [21]. In this type, DRAM is used as the buffer, and the PRAM is the main memory. Therefore, the optimization for this type hybrid main memory is something like the buffer optimization. Moreover, the capacity of the memory is determined only by PRAM.

The right figure in Figure 1 shows the second type hybrid main memory. In this type, PRAM and DRAM have the same function, they all used for storage data. Moreover, they are in the same address space. The performance optimization in this type hybrid main memory is to realize data allocation. Allocate which data in the PRAM medium, and allocate which data in the DRAM data. But the aim for this memory is to allocate frequently write data in the DRAM, and allocate frequently read data in the PRAM. In this way, this type hybrid main memory can take advantages of PRAM in low energy consumption while taking advantages of DRAM in performance.

In this paper, we focus on the second PRAM/DRAM hybrid main memory architecture. Our aim is to reduce average main memory access time to improve main memory performance. Last level cache is used to reduce the gap between CPU and main memory. Optimize last level cache performance can improve memory performance.

Therefore, in this paper, we propose a dynamic hybrid LRU caching scheme (DH-LRU) for PRAM/DRAM hybrid main memory architecture to improve performance.

Firstly, we propose Hybrid LRU policy [19] to make sure the performance improvement. Secondly, we analyze the performance improvement using dynamic hybrid LRU policy.

The rest of this paper is organized as the follows. Section 2 elaborates on essential background and research motivations. We discuss the related work in section 3. Section 4 explains dynamic hybrid LRU (DH-LRU). Section 5 describes experimental methodology and section 6 presents the results of our experiments. Finally, section 7 concludes this paper.

2. Motivation

2.1. Traditional Caching Schemes in PDRAM

As we know, traditional caching schemes just like LRU, FIFO, CFLRU are used widely in traditional memory architecture. They all do their job well in DRAM memory and need not distinguish different cache blocks in different memory medium. But in PRAM/DRAM hybrid main memory architecture that can be a shortcoming. And that has to increase the use of PRAM in PRAM/DRAM if we use these traditional caching schemes directly. In Figure 2, we can know LRU, FIFO RANDOM and CFLRU, four different traditional cache schemes in PDRAM when the N-S calculation program is running with 200 million instructions.

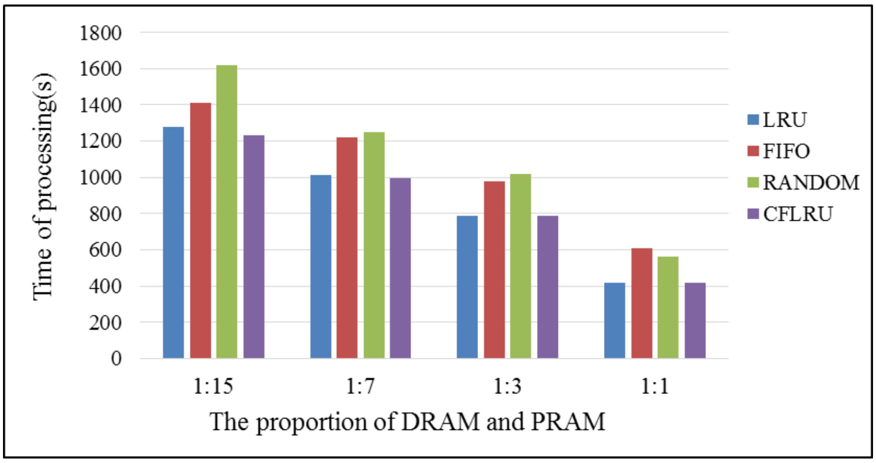

Figure 2. Traditional cache policies in PDRAM

It can't be a judgment if we only have Figure 2 to talk about the disadvantages of traditional caching schemes. But what can be sure is they will not take full advantages of PRAM/DRAM because they don't have enough understanding about PRAM/DRAM. What we need to do is to put forward a new caching scheme to improve this with the specific characteristics of PRAM/DRAM to adapt to different programs, the new caching scheme is named dynamic hybrid LRU(DH-RLU).

3. Related Works

3.1. LRU Caching Scheme

LRU is a simple caching scheme which used widely in both academic and industrial. The basic idea of LRU is, cache blocks are organized by a doubly linked list and the cache block which is most recently used will be insert into the head of list. If the list is full the rear block will be sacrificed if missing happens and keep the list working.

3.2. Clean-First LRU Caching Scheme

CFLRU [7] caching scheme is designed for Flash memory. Because Flash memory has the same characteristic as PRAM. It has limited life and high writing latency. The idea is: much longer time dirty blocks stay in the cache list, much lower replacing cost can be. CFLRU use two doubly caching lists to store cache blocks in order to speed up the replacement. One of the lists is just like an LRU list to store recently used blocks which are dirty, another uses to store clean blocks. The clean blocks can be sacrificed when the missing happens or if the list is full. CFLRU can keep the dirty blocks as long time as it could to reduce the write back times even it costs double overhead of memory space.

3.3. Caching Schemes Based On Frequency Values

Sun G *et al.* propose a caching scheme which based on the writing-frequent values in PRAM memory architecture in [20]. They try to record all the frequency value of wring operation when the program is running. They try to reduce the writing times of same data according to the different frequency values.

Zhang T *et al.* propose a caching scheme mentioned migrating pages in [6] which have high reading frequency values from DRAM to PRAM and high writing frequency values from PRAM to DRAM.

Chen K *et al.* also propose a new memory scheme in PRAM/DRAM in [14]. The main idea is: to make DRAM take more responsibility to do more memory page replacement to reduce the times of PRAM page can be used.

3.4. Shortcomings and Usage

All of the caching schemes which are mentioned above have the same goal, to improve the performance of PRAM/DRAM hybrid memory architecture. But LRU and CFLRU can’t be used in PDRAM/DRAM because they are not capable to distinguish PRAM and DRAM. The caching schemes based on frequency values are applied on memory pages in major. But their ideas are highly reasonable. LRU can be the basic idea of the new caching policy. CFLRU can be the inspiration of reducing writing back times. Others can help notice frequency value of writing and reading to control the new caching policy which means let it adapt to the program dynamically.

4. Dynamic Hybrid LRU Caching scheme

4.1. Principles of Design

In order to adapt to the characteristics of PRAM/DRAM, these principles should be considered when the dynamic hybrid LRU is designed:

a) First of all, DRAM and PRAM should be distinguished in the caching scheme because they have different characteristics.

b) In second, cache blocks which are corresponding DRAM and PRAM should have different operations according to their different characteristics. This is the key to improving the performance of PRAM/DRAM.

c) Thirdly, DRAM cache blocks should take more responsibilities on replacement so that PRAM cache blocks can stay in cache list as long time as possible.

d) What’s more, when the program is running the running time should be splitting into parts for calculating the writing or reading frequency values. And the insert option of DRAM cache block will accord to the frequency values.

e) In the end, how to insert DRAM cache blocks into cache list should accord to the writing and reading frequency value so that dynamic hybrid LRU can adapt to the programs suitably.

4.2. Implementation

The main idea of dynamic hybrid LRU is described in Algorithm1. P means the position which DRAM cache block should be inserted into the cache list. T means threshold which imply the writing frequency will be calculated every t clock cycles. F means the writing frequency in T clock cycles. P and T should be initialized when system start up and P will change according to different writing frequency in every forward threshold, and P1, P2, P3 are defined by different F.

Algorithm 1. Dynamic Hybrid LRU

```
Algorithm 1: Dynamic Hybrid LRU
Output:
Request Cache block
Begin
if DRAM-SPACE(Cache Block address) then
    if cache list is full then
        if ( F is less than 35%)
            move DRAM block to position P1;
        else if ( F is more than 65%)
            move DRAM block to position P3;
        else
            move DRAM block to position P2.
    else
        if list length is larger than P then
            move DRAM block to position P;
        else
            move DRAM block to the tail of list;
        end if
    end if
else
    move PRAM block to the head of list;
end if
End
```

5. Experiment Settings

All experiments are based on an open source architecture simulator which is developed by C program langrage and named Multi2Sim. Multi2Sim provides some traditional caching schemes like LRU, FIFO and RANDOM, they can be called when we set the configure file in the software [15].

5.1. System Configuration

As show in Table 3, Multi2Sim is built in Ubuntu12.04. In order to have a better environment we adding CFLRU and HD-LRU caching schemes manually in the source code. Besides, DRAM and PRAM are bound to a continuous linear address by setting the configure profile.

Table 3. System Configurations

Operating System	Ubuntu 12.04
Simulator	Multi2Sim
Memory Model	DRAM + PRAM
Caching Scheme	LRU/FIFO/RANDOM/CFLRU/HD-LRU

5.2. Configurations

As shown in TABLE 4, total size of memory is 4G, two cache levels are promised in system. Memory is divided into two parts as DRAM and PRAM respectively, and four kinds of memory proportional combinations as 1:15, 1:7, 1:3 and 1:1. L1-cache is fixed to LRU caching scheme. L2-cache is used to test different caching scheme like LRU, FIFO, RANDOM, CFLRU and dynamic hybrid LRU in the experiment.

Table 4. Settings

	Size	Sets	Associate	Block Size
L1-cache	64kb	128	2	256byte
L2-cache	64kb	512	16	256byte
Memory	4G			
D:P	1:15	1:7	1:3	1:1

5.3. Experiment

In the experiment, Mediabench [8] and SPLASH2 [9] are prepared to be the test programs. They collect programs such as N-S fluid calculation, audio encoding, image compress and grid division. There is a few key points and factors of influence:

a) Different proportional combinations of DRAM and PRAM are displayed in TABLE4 as 1:15, 1:7, 1:3 and 1:1.

b) Different positions P in L2-cache with 16 associates are set by P1 = 6, P2 = 10 and P3 = 13.

c) The scale of experiments is unified with running out of 200 million instructions.

d) The threshold is set to 100000, 1000000 and 5000000 clock cycles.

6. Results & Analyses

In 6.1 the comparisons between traditional caching schemes showed on Figure 2 and dynamic hybrid LRU. In 6.2 the effects of position P1, P2 and P3 in cache list is tested and we can see the rules obviously. In 6.3 the dynamic hybrid LRU with 100000, 1000000 and 5000000 clock cycles will be showed. In 6.4 the overhead on both software and hardware can be presented. In 6.5 the energy consumption is calculated and displayed.

6.1. Performance Improvement

The performances of traditional caching scheme in PRAM/DRAM is showed in Figure 2 which was tested by N-S fluid calculation program. We also test dynamic hybrid LRU in the same situation and the performance can be seen on Figure 3.

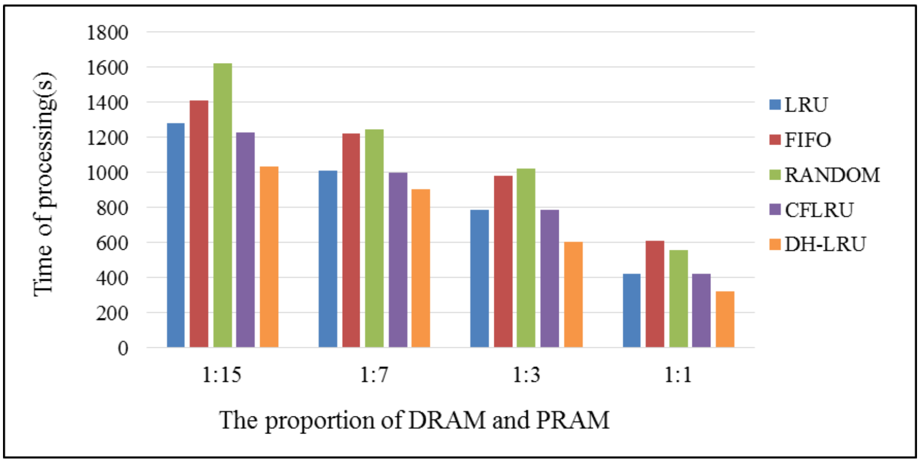

Figure 3. DH-LRU vs Traditional Schemes in PRAM/DRAM

Dynamic hybrid LRU performed much better than any other traditional cache schemes. As we said, dynamic hybrid LRU can distinguish different mediums in PRAM/DRAM and also adapt to the program by the writing and reading frequency. This is the first step to prove our theory.

6.2. The Effects of Different Proportion and Position P

Different proportion of DRAM and PRAM will have different performance in PRAM/DRAM. And different position P also present different results. These are two most important factors which impact performance of DH-LRU a lot. As show in Figure 4, we test the performance with four different proportions and three different position P by using N-S Fluid Calculation just the same program as used in Figure 1 and Figure 2.

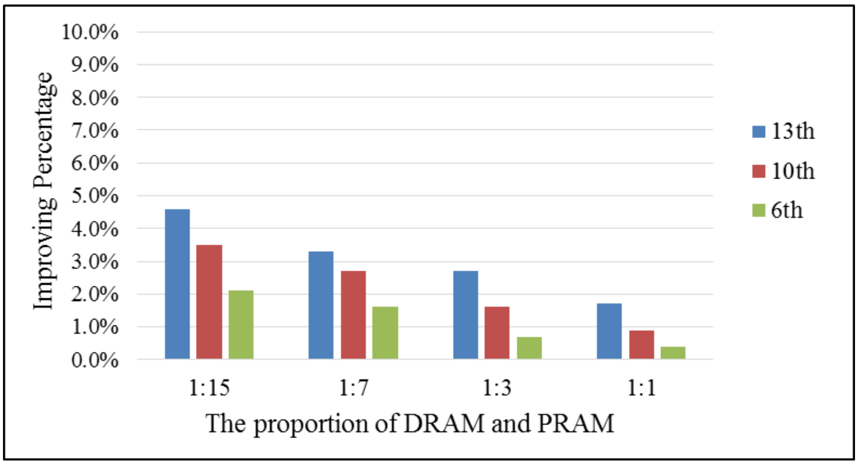

Figure 4. N-S Fluid Calculation in PRAM/DRAM with Hybrid-LRU

Figure 4 shows the static situation without any detection to the writing frequency which named Hybrid-LRU. The length of caching list is 16, we can see the rules clearly. In the same proportion of DRAM and PRAM, more closing the position P to the rare of the list is, more improvement percentage will present. It could speed up the program of running by 4.7% in maximum. Because DRAM could be replaced much more times if it is closed to the rear of LRU list and that could help to reduce the using of PRAM blocks.

But we can also notice that when proportion of DRAM becomes larger the improvement percentage is reduced. The reason is DRAM has much lower writing latency than PRAM, more DRAM in the DRAM/PRAM hybrid memory architecture can make the whole memory much faster but also more power consumption. So the DRAM/PRAM is fast enough at the beginning that may cause more difficulties on improving it. We do not suggest too much DRAM is designed in the DRAM/PRAM hybrid memory because that will not have enough PRAM to use and the power consumption could be much higher.

We get similar results on other different programs as Figure 5. Image Compressing, Audio Encoding and Grid Division program present the same tendency when the position P is changed among 6, 10 and 13 in the proportion of 1:15. These are the results of Hybrid-LRU which just has static position P. Dynamic hybrid LRU(DH-LRU) will control the change of position P according to the difference of writing frequency when the programs are running. We want to show the performance simply and clearly so we will just use the proportion of 1:15 in the following pages to introduce DH-LRU.

6.3. The Effects of Different Threshold

The threshold could impact the frequency of writing detection in DH-LRU. We set t equals 100000, 1000000 and 5000000 clock cycles as the threshold which means when the programs running up to t clock cycles there should be a calculation of its writing frequency. If the writing frequency is in some defined range then the position P will be changed as correspondence rules. Different threshold means the different frequency of calculations of writing frequency so that the small threshold is, the more often P changes. And the appropriate threshold could also decide the performance of DH-LRU just as position P and the property of DRAM and PRAM in the hybrid memory architecture.

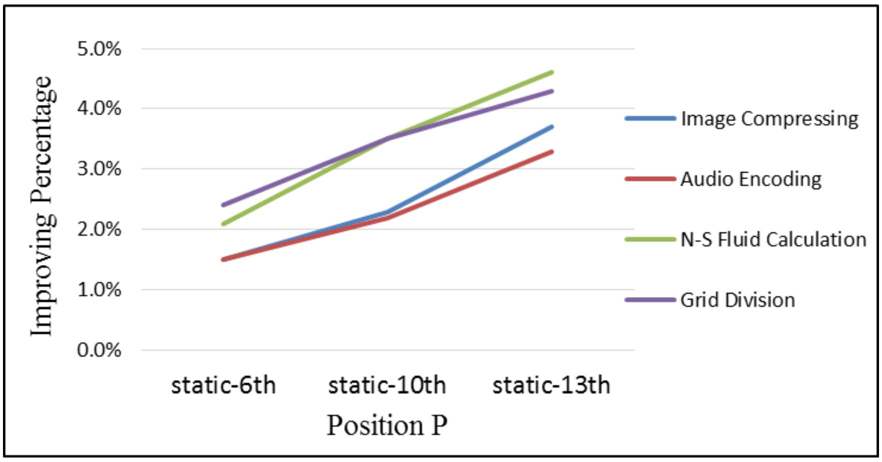

Figure 5. Test Programs in PRAM/DRAM with Hybrid-LRU

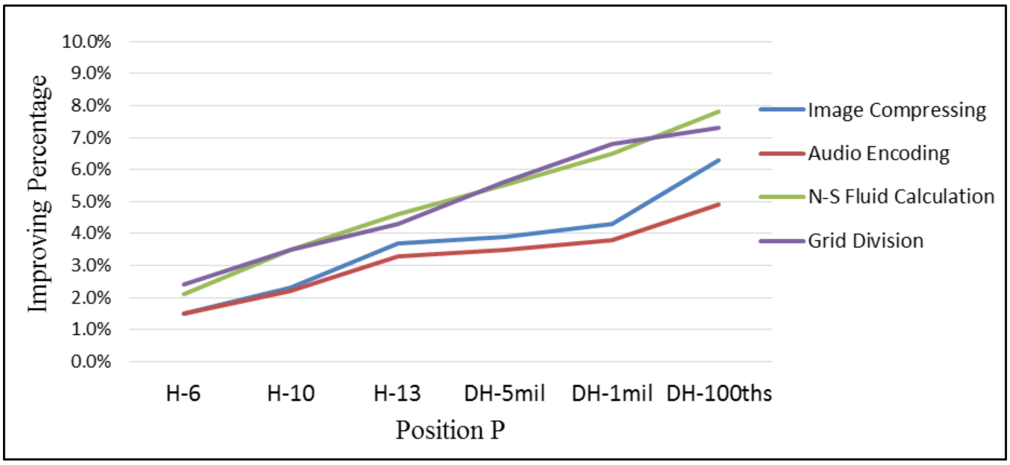

Figure 6. Test Programs in PRAM/DRAM with DH-LRU

As shown in Figure 6, we put the Hybrid-LRU data and DH-LRU data together to see the performance. H-6 means position P is fixed on the 6th of the caching list and the DRAM cache block will be inserted into this position, same as H-10 and H-13. DH-5mil means threshold is set to 5 million clock cycles, same as DH-1mil, DH-100ths means threshold is set to 100000 clock cycles to calculate the writing frequency in PRAM/DRAM. Fortunately, four programs present the same tendency. Even we set a larger threshold as 5 million clock cycles for DH-LRU, it performs better than Hybrid-LRU in any static position P because dynamic settings help DH-LRU adapt to the writing and reading in any program. If we set a smaller threshold as 100000 clock cycles it could perform better again. But it doesn’t mean the smaller threshold can bring better performance. If the threshold is too small it could make position P changing too frequently which may cause a lot unnecessary calculation and power consumption.

6.4. Utilization of PRAM

DH-LRU change position P in cache list dynamically to improve the performance of PRAM/DRAM. The main idea of DH-LRU is inherit from Hybrid-LRU. Both of DH-LRU and Hybrid-LRU are to decrease the use of PRAM and increase the use of DRAM.

As shown in Figure 7, utilization percentage of PRAM in N-S Fluid Calculation in four different proportions of DRAM and PRAM is present and position P is set to 13. When the DRAM proportion get larger the use of PRAM becomes smaller. We can see DH-LRU has much better performance and reduce much amount usage of PRAM than Hybrid-LRU and other traditional caching schemes in PRAM/DRAM. That is because DH-LRU is not just put DRAM cache blocks as closing to the rear of cache list as it could. If the program just need to do read then the DRAM cache block also allow to stay in the list longer. In Hybrid-LRU the static position P has already reduces the utilization of PRAM to 88.2% at most and 94.5% in average [19]. In DH-LRU the utilization of PRAM has reduced to 72% when the proportion is 1:15.

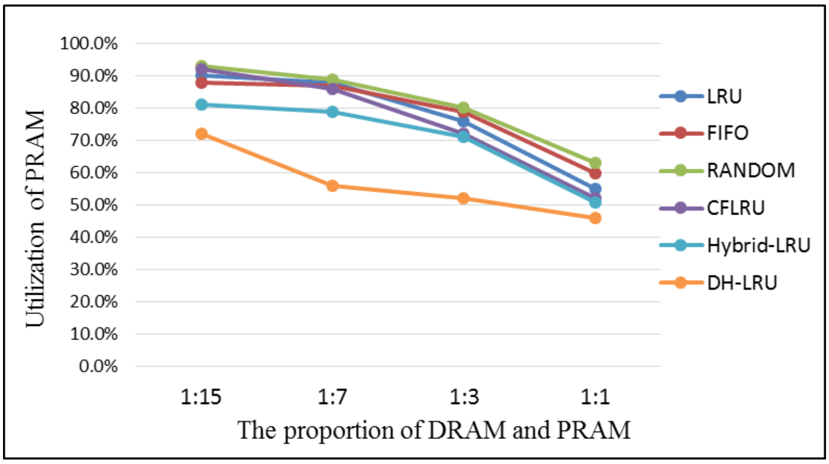

Figure 7. Test Programs in PRAM/DRAM with DH-LRU

6.5. Overhead of DH-LRU

As we know DH-LRU could bring better performance than Hybrid-LRU and other traditional caching schemes. It needs to distinguish DRAM and PRAM in PRAM/DRAM hybrid memory architecture and calculate the writing frequency in every threshold. To describe the overhead [22] of HD-LRU we will describe on both hardware and software. From a hardware standpoint, position P and the writing frequency F may take a certain amount of hardware space. The location where a cache block should be inserted is up to F and P, and the location of a cache block in the cache list is up to memory address [23, 24].

Physical Address	tag	set	way
	t	s	w

Figure 8. The structure of a memory address

Figure 8 shows the structure of a memory address in 32 bit. It is divided into 3 parts as tag, set and way [16]. These three parameters are used to mapping the cache block in cache system. Fortunately, set, way and tag are provided in Multi2Sim source code [10, 11], which means we can get the physical address in memory with these three parameters with Equation 1. Algorithm 2 is the specific methods.

$$PA = tag << (s + w) + set << w + way \quad (1)$$

In software position P and the writing Frequency F of writing frequency is major features of DH-LRU. 16 ways are configured in the experiment so that 8 bits are required to address as a flag of position P and F. 512 sets with 16 ways need 512 bytes total space to store position Ps and frequency F.

6.6. Energy Consumption of DH-LRU

A lot of schemes are proposed [12, 25] for saving the energy PRAM/DRAM. In PRAM/DRAM hybrid memory architecture DRAM cost a lot energy because it needs to refresh to keep data. So we suggest less DRAM proportion in PRAM/DRAM because it has much better performance on energy consumption.

Algorithm 2. Back to Physical Address

```
Algorithm 2: Back to Physical Address
Output:
Cache Block (tag, set, way)
Begin
if (tag, set, way) is valid then
    PA = tag << (s + w) + set << w + way;
        if PA belongs DRAM region then
            Cache Block belongs DRAM region;
        else if PA belongs PRAM region then
            Cache Block belongs PRAM region;
        else
            Wrong Space;
        end if
else
    Wrong Space;
end if
End
```

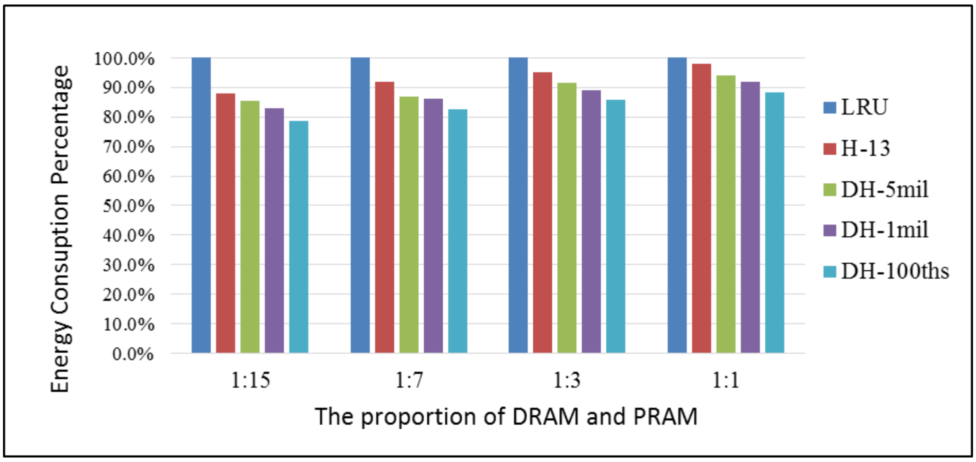

Figure 8. Energy Consumption in different proportion with DH-LRU

By the way, PRAM has only 1/26 idle power consumption of DRAM even though the average consumption of writing and reading of PRAM is three times as DRAM because of the heating temperature [13]. Figure 7 presents DH-LRU reduces the utilization of PRAM to 72% at most in the proportion of 1:15 and position P in 15. Figure 8 shows the reduction of the average of consumption of writing and read for PRAM/DRAM with DH-LRU. We assume the energy consumption of LRU as 100% and compare with hybrid LRU (HLRU) in position P is 13 and dynamic hybrid LRU (DHLRU) with different threshold. We can see DHLRU save more energy consumption than HLRU and has better performance in 1:15 than other proportion.

7. Conclusion

This paper presents dynamic hybrid LRU (DHLRU) which is inherit from hybrid LRU (HLRU). The main idea of DHLRU is just as same as HLRU: take advantages of DRAM cache blocks to make up the disadvantages of PRAM cache blocks. Just set P as a fixed position is not good enough for different kind of programs. DHLRU set a threshold to calculate the front writing frequency to predict the writing situation for the next part of program. As we tried 5000000, 1000000, 100000 clock cycles as the threshold we found the smaller threshold value brings better performance because position P will change dynamically according the writing frequency to adapt to the program and the result could be more accurate. There's no doubt that DHLRU has much better performance than HLRU because of its flexibility.

Acknowledgments

This work was supported by the National Science Foundation of China under grants (No. 61602137, No. 61572163). Zhejiang provincial Natural Science Foundation (No. LQ14F020011). National science and technology supporting plan (2014BAK14B00).

References

[1] Kim K, Jeong G. Memory technologies in the nano-era: challenges and opportunities[J]. Technology, 2011, 120(180): 240.

[2] S. Electronics. (2010, Apr.). Samsung ships industry's first multichip package with a PRAM chip for handset[Online]. Available: http://www.samsung.com.

[3] Choi Y, Song I, Park M H, *et al*. A 20nm 1.8 V 8Gb PRAM with 40MB/s program bandwidth[C]//Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International. IEEE, 2012: 46-48.

[4] Burr G W, Breitwisch M J, Franceschini M, *et al*. Phase change memory technology[J]. Journal of Vacuum Science & Technology B, 2010, 28(2): 223-262.

[5] Simpson R E, Fons P, Kolobov A V, *et al.* Interfacial phase-change memory[J]. Nature nanotechnology, 2011, 6(8): 501-505.

[6] Zhang T, Xing J, Zhu J, *et al*. Exploiting Page Write Pattern for Power Management of Hybrid DRAM/PRAM Memory System[J]. JOURNAL OF INFORMATION SCIENCE AND ENGINEERING, 2015, 31(5): 1633-1646.

[7] Ji X, Zeng F. Flash-aware Virtual Memory System for Consumer Electronics[J]. International Journal of Multimedia and Ubiquitous Engineering, 2015, 10(8): 329-338.

[8] Wang Y, An H, Liu Z, *et al*. Speeding Up Mediabench in Loop&Procedure Level Speculation[J]. International Journal of Advancements in Computing Technology, 2012, 4(11).

[9] Drummond M. Power and Performance Characterization of Splash2 Benchmarks on Heterogeneous Architecture[J]. 2015.

[10] Guangjie Han, Liangtian Wan, Lei Shu, Naixing Feng, Two Novel DoA Estimation Approaches for Real Time Assistant Calibration System in Future Vehicle Industrial, IEEE Systems Journal, 2015. DOI: 10.1109/JSYST.2015.2434822

[11] Zhang J, Liao X, Jin H, *et al*. An Optimal Page-Level Power Management Strategy in PCM–DRAM Hybrid Memory[J]. International Journal of Parallel Programming, 2015: 1-13.

[12] Guangjie Han, Jiawei Shen, Li Liu, Lei Shu, BRTCO: A Novel Border Line Recognition and Tracking Algorithm for Continuous Objects in Wireless Sensor Networks, IEEE Systems Journal, 2016, DOI: 10.1109/JSYST.2016.2593949

[13] G. Jia, G. Han, J. Jiang, J. Rodrigues. PARS: A Scheduling of Periodically Active Rank to Optimize Power Efficiency for Main Memory. Jounal of Network and Computer Application. Vol. 58, pp. 327-336, 2015.

[14] Guangjie Han, Wenhui Que, Gangyong Jia, Lei Shu. An Efficient Virtual Machine Consolidation Scheme for Multimedia Cloud Computing, Sensors, Vol.16, No.2, Article 246, 2016

[15] Chen K, Jin P, Yue L. A novel page replacement algorithm for the hybrid memory architecture involving PCM and DRAM[M]//Network and Parallel Computing. Springer Berlin Heidelberg, 2014: 108-119.

[16] Yi Ou, Theo Harder, Clean First or Dirty First? A Cost-Aware Self-Adaptive Buffer Replacement Policy, IDEAS10 2010, August 16-18.

[17] Dong J, Zhang L, Han Y, *et al*. Wear rate leveling: lifetime enhancement of PRAM with endurance variation[C]//Proceedings of the 48th Design Automation Conference. ACM, 2011: 972-977.

[18] Ping Zhou, Bo Zhao, Jun Yang. A durable and energy efficient main memory using phase change memory technology[C]. ISCA 09 Proceedings of the 36th Annual International Symposium on Computer Architecture, 2009:14-23.

[19] Hongtianchen Xie, Gangyong Jia, *et al*. Hybrid-LRU Caching Scheme for PDRAM Hybrid Memory Architecture in Cloud Computing[C]. Proceedings of the Identification, Information and Knowledge in the Internet of Things, 2016.

[20] Sun G, Niu D, Ouyang J, *et al*. A frequent-value based PRAM memory architecture[C]//Proceedings of the 16th Asia and South Pacific Design Automation Conference. IEEE Press, 2011: 211-216.

[21] Dhiman G, Ayoub R, Rosing T. PDRAM: a hybrid PRAM and DRAM main memory system[C]//Design Automation Conference, 2009. DAC'09. 46th ACM/IEEE. IEEE, 2009: 664-669.

[22] Liu Y, Geurts J, Point J C, *et al*. Dynamic adaptive streaming over CCN: a caching and overhead analysis[C]//Communications (ICC), 2013 IEEE International Conference on. IEEE, 2013: 3629-3633.

[23] Nanavati M, Spear M, Taylor N, *et al*. Whose cache line is it anyway?: operating system support for live detection and repair of false sharing[C]//Proceedings of the 8th ACM European Conference on Computer Systems. ACM, 2013: 141-154.

[24] Prabhakar R, Srikantaiah S, Kandemir M, *et al*. Adaptive multi-level cache allocation in distributed storage architectures[C]//Proceedings of the 24th ACM International Conference on Supercomputing. ACM, 2010: 211-221.

[25] Shin D J, Park S K, Kim S M, *et al*. Adaptive page grouping for energy efficiency in hybrid PRAM-DRAM main memory[C]//Proceedings of the 2012 ACM Research in Applied Computation Symposium. ACM, 2012:395-402.

Authors

Yongjian Ren is currently a Professor of Department of Computer Science at Hangzhou Dianzi University, China. He received his Ph.D. degree in Department of Computer Science from Zhejiang University, Hangzhou, China. His current research interests are storage.

Hongtianchen Xie is now the graduate student in Hangzhou Dianzi University, China. His current research interests are power management, operating system, cache optimization, memory management.

Gangyong Jia is currently an Assistant Professor of Department of Computer Science at Hangzhou Dianzi University, China. He received his Ph.D. degree in Department of Computer Science from University of Science and Technology of China, Hefei, China, in 2013. He has published over 40 papers in related international conferences and journals. He has served as a reviewer of IEEE transactions on Computers, Microprocessors and Microsystems, and so on. His current research interests are power management, operating system, cache optimization, memory management. He is a member of IEEE.

Jilin Zhang is currently an associate Professor of Department of Computer Science at Hangzhou Dianzi University, China. He received his Ph.D. degree in Department of Computer Science from University of Science and Technology of China, Beijing, China. He has published over 50 papers in related international conferences and journals. His current research interests are cloud computing, parallel computing.

Yuyu Yin is currently an associate Professor of Department of Computer Science at Hangzhou Dianzi University, China. He received his Ph.D. degree in Department of Computer Science from Zhejiang University, Hangzhou, China. He has published over 30 papers in related international conferences and journals. His current research interests are cloud computing, service computing.

Jian Wan is currently a Professor of Department of Information and Electronic Engineering, Zhejiang University of Science and Technology, China. He received his Ph.D. degree in Department of Computer Science from Zhejiang University, Hangzhou, China. He has published over 80 papers in related international conferences and journals. His current research interests are cloud computing, multimedia, and so on.