Virtual Page Behavior based Page Management Policy for Hybrid Main Memory in Cloud Computing

Jie Huang[1,2], Xuan Zhang[1], Guangjie Han[3], Gangyong Jia[1,2], Huizi Liyou[1], Jian Wan[4]
[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] Department of Information & Communication Systems, Hohai University
Changzhou, 213022, China
[4] School of Information and Electronic Engineering, Zhejiang University of Science and Technology
Hangzhou, 310023
huangjie@hdu.edu.cn; gangyong@hdu.edu.cn

Abstract

A new generation memory, Non-Volatile Memory (NVM), such as Phase-Change Memory (PCM), has been adopted together with DRAM in the main memory to form the hybrid main memory for low energy consumption and high capacity. The biggest challenge of hybrid memory is how to decrease the average memory access cost for the higher cost of NVM's read/write operation. Currently, most researches are based on migration. However, the page migration itself is a high cost operation. And the migration based policy produces many migration operations, which induces high cost in memory access. Therefore, in order to decrease the cost, we present a virtual page behavior based page management policy (VBPM) in this paper. According to the virtual pages' behavior, we allocate virtual pages into DRAM or PCM physical pages correspondingly. The whole process is migration independent. The experimental results show our VBPM decreases the average memory access time by 24%, moreover, VBPM improves real-time performance in critical path.

Keywords

hybrid memory, PCM, memory access, virtual page

1. INTRODUCTION

In the era of big data and cloud computing, the needs for large energy-efficient main memory are increasing seriously. Compared with DRAM, Phase-Change Memory (PCM) is non-volatile and expected to have higher storage density in the future [1, 2]. Moreover, PCM is byte-addressable and supports random access, which is becoming a great substitute for main memory.

However, two problems make it difficult to replace DRAM in current computer systems. First, the write latency of PCM is about 6 to 10 times larger than that of DRAM. Second, PCM cells have limited write endurance [3]. Therefore, a more practical way to utilize PCM in memory system is to utilize advantages of both DRAM and PCM to construct hybrid memory architecture [2], demonstrating in figure 1.

In such hybrid memory architecture, DRAM and PCM are using a single physical address space, which can take advantages of the high density, low power consumption and non-volatile of PCM and the high efficiency in memory access of DRAM [16]. However, one of the most challenges of hybrid memory is the average memory access time. How to decrease the average memory access time, which means how to decrease the number of write operations in the PCM, and at the same time, decrease read operations in the DRAM [15].

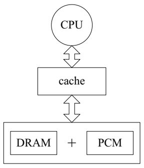


Fig. 1 hybrid memory architecture

Recently, a large number of researches have proposed to decrease the average memory access time or prolong the PCM lifetime. Most of them are based on the page migration. They set a threshold in advance in the memory management [17]. When a physical page's write operations exceed the threshold in the PCM, migrate it to the DRAM. Migrate it to DRAM can decrease the average memory access time for the high write cost in the PCM. When a physical page's read operations exceed the threshold in the DRAM, migrate it to the PCM [19]. Migrate it to PCM can decrease the average memory access time for reducing competition to the write intensive pages in DRAM [18]. However, migration itself is a high cost operation [5-8].

In order to reduce the overhead of migration and improve efficiency of memory access in DRAM/PCM hybrid memory, we propose a virtual page behavior based page management policy (VBPM) in this paper.

Overall, the contributions of our work are as follows:

1) In order to decrease the migration cost, our VBPM has no migration, therefore, the cost is reduced.
2) Our VBPM predicts each page's memory read/write behavior based on the results after running on simulator. According to the virtual pages' behavior, map these virtual pages into the DRAM or PCM physical pages. In this way, the most write operations are in the DRAM and most read operations are in the PCM, which is efficiency.
3) We use different workloads to compare VBPM with the migration based memory management policy. The experimental results show that VBPM decreases the average memory access time by 24%, moreover, VBPM improves real-time in critical path.

978-1-5090-5696-5/16 $31.00 © 2016 IEEE
DOI 10.1109/MSN.2016.27

2. RELATED WORKS

For the architecture of DRAM/PCM hybrid main memory, there are two main streams, page heat and page intensive. For the algorithm of page heat partition, Lee et al proposed CLOCK with dirty bits and write frequency (CLOCK-DWF) [9]. CLOCK-DWF allocates memory pages according to the request type of the data that has not been hit. Read request page is allocated into the PCM memory, write request page is allocated into DRAM memory. When a write requests miss or it hits in PCM, the page would be moved to DRAM.

For the page intensive partition method, Seok et al [10, 11] proposed LRU with prediction and migration (LRU-WPAM). LRU-WPAM monitors all physical pages and obtains the pages' weight value according to the page's read and write history. This weight value means the read/write intensive of the page. DRAM owns a read-intensive list and PCM owns a write-intensive list. When the weight value of the page in read-intensive list exceeds a certain threshold, it will be migrated to the PCM. In the contrary, the page in write-intensive will be migrated to the DRAM, if the weight value exceeds a certain threshold.

Different to the above algorithms, Maintain-hit-ratio LRU (MHR-LRU) [12] algorithm maintains a DRAM write list (DWL), only in write hit, the list will be changed. The write hit page be moved to the MRU side of the DWL, it makes the page in the side of MRU has more frequently writing. In contrast, the page in LRU side has less write operations. If a page in the MRU side of the LRU at the moment, and also in the LRU side of the DWL, meanwhile a write request is coming, the page in the LRU side of the DWL will be migrated to the PCM to make a space for new write request.

Access-pattern-prediction-based LRU (APP-LRU) [13] algorithm maintains an additional historical metadata table, which records the write and read history of the page accessing the disk. According to the metadata, APP-LRU predicts the page's read/write intensive. Allocate the read-intensive page to the PCM and the write-intensive page to the DRAM.

Above methods' data are obtained by monitoring the physical page's read or directly allocate the page according to the single step of request, depend on the migration to fix up the wrong partition. Migrate between two mediums need the help of the transfer-buffer, accessed the transfer-buffer cost 0.06nJ per 64B data [14].

3. MOTIVATIONS

In the section 3, we have analyzed four of the most representatives migration based algorithms' disadvantages. In this section, we prove our above analysis though experimental results, containing migration number analysis, overhead analysis, and DRAM utilization analysis.

3.1 Migration Analysis

Figure 2 shows the migration numbers of these four algorithms. The results are based on running benchmark OPENFILE. Among these four algorithms, LRU-WPAM has the most migrations, reached to 4284, APP-LRU has the least migrations, almost 105. These results are consistent with the analysis in section 3.1.

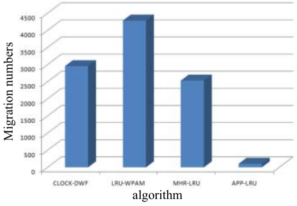


Fig. 2 the migration numbers of these four algorithms

In order to analyze the migrations in more detail, figure 3 shows LRU-WPAM's detailed migration data. In this figure, the x-axis represents page types of different migration numbers, 0 represents page type without migration pages, 1~10 represents page type of pages having migration numbers among 1 to 10. The y-axis represents the percentage of different page types. Most pages have no migration, reached to 96.1%. Pages having 1 to 10 migrations occupy 3.69%, pages having 11 to 100 migrations occupy 0.12%, and page having more than 100 migrations occupy 0.09%. Therefore, the migration clusters in a few pages. Especially, one page reaches 3254 migrations in LRU-WPAM for its special in migration trigger mechanism.

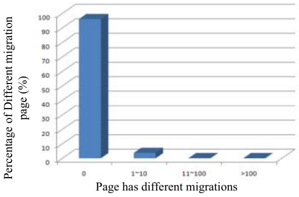


Fig. 3 LRU-WPAM's detailed migration data

3.2 Overhead Analysis

The more write operations in the PCM memory, the more overhead will occur for high cost in PCM write. Therefore, we analyze the write operation numbers of these four algorithms.

Figure 4 shows the four algorithms' write operation numbers in PCM memory when running the same benchmark. The less migration algorithms APP-LRU and MHR-LRU have more write operation numbers in PCM memory. So many write operations in MHR-LRU are mainly from two parts: 1) Some pages of few write operations are migrated to the PCM memory, which induce write operations in PCM memory; 2) There is no migration from PCM to DRAM, which means frequently write pages keep in the PCM. Therefore, induce many write operations.

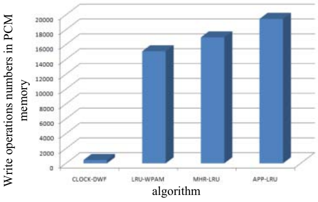


Fig. 4 the four algorithms' write operation numbers in PCM memory

Combined figure 2 and 4, we can see migration based algorithms have high overhead, high migrations or high write operations in PCM memory.

3.3 DRAM Utilization Analysis

The number of the read times in DRAM and the number of the whole access can be used to evaluate the DRAM utilization. The DRAM utilization reflects the efficiency of DRAM memory. If the DRAM utilization is too high, the efficiency of DRAM will decrease for increasing competition.

Figure 5 shows the four algorithms' read operation numbers in DRAM memory when running the same benchmark. CLOCK-DWF has the most read operations in DRAM memory, reached 100988, which means the CLOCK-DWF is most dependent to the DRAM memory.

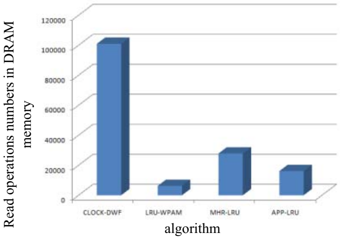


Fig. 5 the four algorithms' read operation numbers in DRAM memory

Figure 6 shows the formed of total memory access numbers of four algorithms. pcmr represents read operations in PCM memory, pcmw represents write operations in PCM memory, dramr represents read operations in DRAM memory, dramw represents write operations in DRAM memory respectively. Through this figure, we can see all parts consist of memory access. The more pcmw, there is more overhead. The more dramr, there is more dependent to the DRAM.

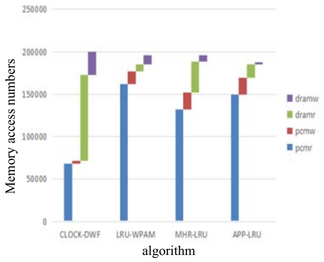


Fig. 6 the formed of total memory access numbers of four algorithms

4. A VIRTUAL PAGE BEHAVIOR BASED PAGE MANAGEMENT POLICY (VBPM)

4.1 Page Allocation Algorithm

The most important of the memory management in DRAM/PCM hybrid memory system is the page allocation algorithmIn order to implement page allocation, some thresholds need to be determined, such as maximum write numbers, *W*, the ratio of read operations to write, α. A good page allocation algorithm will map frequently write pages into DRAM memory and map mostly read pages into PCM memory.

In order to evaluate both performance and overhead of migration based memory management policy at currently, we present an evaluation model. The evaluation model is to judge the correctness of the page allocation algorithm. In this paper, when a page's ratio of read operations to write exceeds α and the maximum write number doesn't exceed *W*, then the correct way is to place the page into the PCM. Otherwise, when a page's maximum write number exceeds *W*, then the correct way is to place the page into the DRAM. If a page allocation algorithm doesn't satisfy above condition, then it's an error.

The key is how to determine *W* and α. α can be set based on experiences data, however, *W* is based on each process.

Therefore, firstly, we determine W through formula (1)

$$W = c * t / T_e \qquad (1)$$

W means the maximum write number threshold. T_e means expected running time of the PCM. *c* represents maximum write times PCM can maintain correct. And *t* is the running time of the single process.

Algorithm 1: Page Allocation algorithm

Input:

(1) a page request q

(2) PL is the LRU list of the PCM, DL is the LRU list of the DRAM. L is the LRU list of the Hybrid Main Memory

(3) HB is the access history map, containing virtual page address and the numbers of read and write

(4) W is the maximum write time threshold,α is the threshold of the ratio of read to write

Output:

a reference to the requested page

```
Begin
if HB[q].write > W then
    if there is a free physical page in DRAM then
        return a reference to the free physical page;
    else
        return a reference to LRU position of DL;
else
    if HB[q].read/HB[q].write > α then
        if there is a free physical page in PCM then
            return a reference to the free physical page;
        else
            return a reference to LRU position of PL;
    else
        return a reference to LRU position of L;
End
```

According the results of the Algorithm 1, we can get the most suitable page map for each process.

4.2 Detail of the Policy

Figure 7 shows the framework of our VBPM. VBPM mainly contains three parts: first, in order to predict each page's memory access behavior, we run processes in the simulator,which runs only once and can be run in the cloud. After running in the simulator, get each virtual page's memory read/write behavior. Second, analyze the DRAM/PCM hybrid memory to determine the threshold for page allocation. Third, according to both virtual page's behavior and threshold, map each virtual page into DRAM or PCM physical page by Algorithm 1.

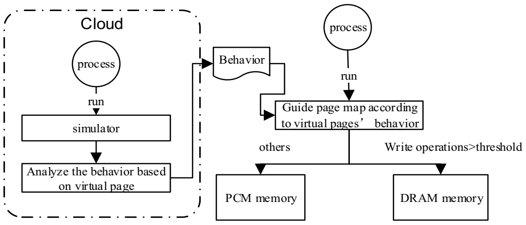

Cloud
process
run
simulator
Analyze the behavior based on virtual page
Behavior
process
run
Guide page map according to virtual pages' behavior
others
Write operations>threshold
PCM memory
DRAM memory

Fig. 7 the framework of our VBPM

The local and global access behavior of the process are similar, but partition page by local access behavior has errors and delays. The errors lead to migration, and the delays make the error page can't be moved to more suitable medium immediately. If it can be judged in suitable medium while the page is allocating, there is not any migration during the process running can also guarantee the PCM utilization and less write operation in PCM. So it not only can make full use of system performance, but also ensure the lifetime of the system.

We present the method VBPM to partition page according to the virtual page access behavior in order to solve the problems the migration algorithm brought. Combined with the global virtual address memory access behavior, we can know that the process of running all the memory operations, including the initialization of the page, read operations, write operations, as well as the execute operation. Analyzing these operation grouped by page can obtain the single page's behavior during the whole running time of process. According the different behaviors of the page, we can partition them to the different mediums.

The simulator can be run in the cloud. And each process only runs once in the cloud to get the behaviors. So, the cost of running simulator is low for the process that runs multiple times.

Compared with LRU-WPAM, VBPM has allocated the suitable medium for each page, avoid the reciprocating migration. VBPM has less write operations in PCM than APP-LRU and MHR-LRU. And there is less dependence on DRAM compared with the CLOCK-DWF, avoid the waste of the PCM.

5. RESULTS & ANALYSES

In this section, we used Multi2Sim simulator to emulate two test programs, obtaining the trace of memory access during the process running. Moreover, simulate these traces using different algorithm including VBPM, to prove VBPM improve efficiency of memory access.

5.1 Experimental Setup

We build the Multi2Sim at the first. We get the memory access trace of the process by modify the Multi2Sim's source code. Each line of the trace consists of a virtual address and an operation code. Then we grouped this address per 4KB as a page, transfer the virtual address behavior to virtual page behavior. In order to comprehensively analyze both the advantages and disadvantages of different algorithms, we capture a large number of procedure's access trace, and two kinds of representative trace were selected as the experimental input.

Table 1 Information of the experiment

Trace	Used page	Write request number	Read request number	Running time
FFT	213	120486	294511	3.35
OPENFILE	3384	24707	166035	1.52

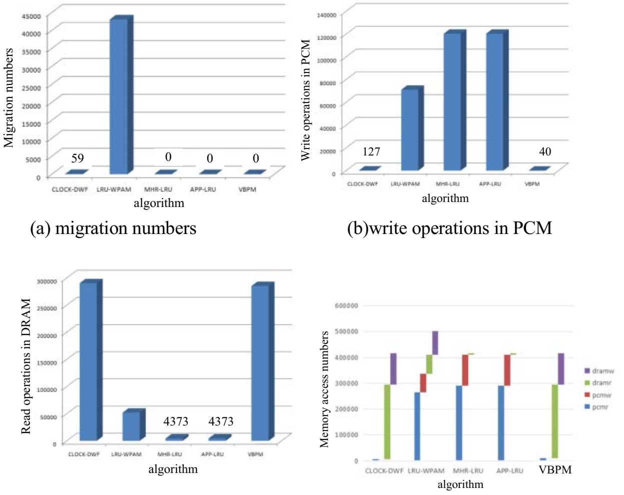
FFT trace is the simulating result of the Splash2/fft with the parameter -m10 -p1 -n65535 -l4. OPENFILE trace is the result of a procedure that open a file whose size is 10MB.

Because the trace of experiment is single process, would not cost too many memories. To make the result more intuitive, we set the main memory size 400 pages, 1.6Mb. The ratio of PCM to DRAM is 3 to 1.FFT trace used less than this, OPENFILE trace used closely.

The VBPM algorithm's parameters are listed as follow: expected running times T_e is 3153600,which means hybrid main memory run one year without interruption, the maximum times PCM can be written is 10^8, the threshold of the ratio of read to write is 2.

5.2 Experimental Results

Figure 8 shows the results of FFT trace. Because FFT trace uses less pages than whole main memory, MHR-LRU and APP-LRU have no migration. VBPM has least write in PCM with the 0 migration. VBPM has the least number of memory accesses. Because of the page write frequently also has a lot of reads, the performance of the VBPM is similar as the CLOCK-DWF in this trace.



(a) migration numbers

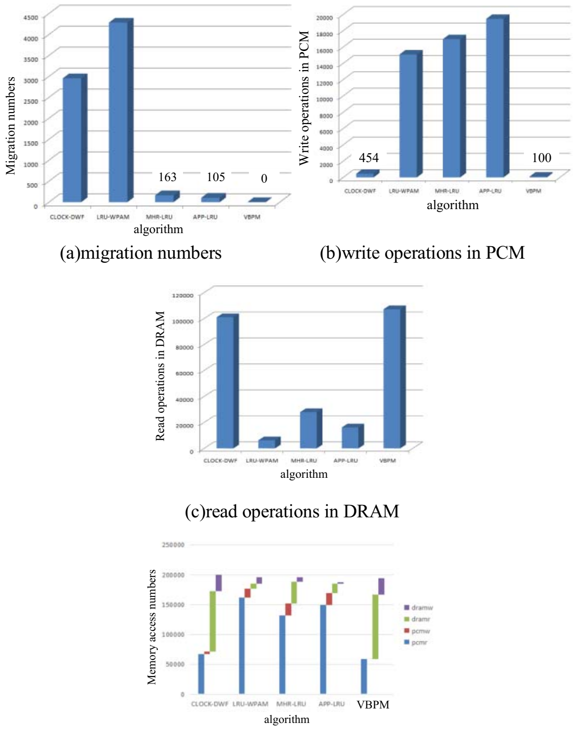
(b)write operations in PCM

(c) read operations in DRAM

(d)total memory access

Fig. 8 Simulation results of the trace FFT

Figure 9 shows the simulation results of the trace OPENFILE. When the used page is more than the main memory pages, means page replace frequently, the migration cause by CLOCK-DWF has a dramatic increasing. But the VBPM with the premise 0 migration has least PCM write and least whole access.



(a)migration numbers

(b)write operations in PCM

(c)read operations in DRAM

(d)the formed of total memory access numbers of four algorithms

Fig.9 Simulation results of the trace OPENFILE

6. CONCLUSION

In order to reduce the overhead of migration and improve efficiency of memory access in DRAM/PCM hybrid memory, we propose a virtual page behavior based page management policy (VBPM) in this paper. In VBPM, firstly, in order to predict each page's memory access behavior, we run processes in the simulator. After running in the simulator, get each virtual page's memory read/write behavior. Secondly, analyze the DRAM/PCM hybrid memory to determine the threshold for page allocation. Thirdly, according to both virtual page's behavior and threshold, map each virtual page into DRAM or PCM physical page. Through our VBPM, there is no migration. Moreover, the most write operations are in the DRAM and most read operations are in the PCM. Therefore, improve efficiency of the memory access.

7. ACKNOWLEDGMENT

This work was supported by the National Science Foundation of China under grants (No. 61602137, No. 61572163, No. 61572172, No. 61402140). Zhejiang provincial Natural Science Foundation (No. LQ14F020011). The Fundamental Research Funds for the Central Universities (No. 2016B10714), PAPD and CICAEET. Zhejiang province public welfare Technology Research Program Grant (No. 2015C33064).

8. REFERENCES

[1] Park H, Yoo S, Lee S. Power management of hybrid DRAM/PRAM-based main memory[C]//Proceedings of the 48th Design Automation Conference. ACM, 2011: 59-64.

[2] Wu Zhangling,Jin Peiquan,Yue Lihua, et al. A Survey on PCM Based Big Data Storage and Management[J]. Journal of Computer Research and Development, 2015, 52(2): 343-361.

[3] Chen S, Gibbons P B, Nath S. Rethinking Database Algorithms for Phase Change Memory[J]. Cidr, 2011:21-31.

[4] Zhou P, Zhao B, Yang J, et al. A durable and energy efficient main memory using phase change memory technology[J]. Acm Sigarch Computer Architecture News, 2009, 37(3):14-23.

[5] G. Jia, G. Han, J. Jiang, J. Rodrigues. PARS: A Scheduling of Peri odically Active Rank to Optimize Power Efficiency for Main Mem ory. Jounal of Network and Computer Application. Vol. 58, pp. 32 7-336, 2015.

[6] G. Jia, G. Han, D. Zhang, L. Li, L. Shu. An Adaptive Framework for Improving Quality of Service in Industrial Systems. IEEE ACCESS. Vol. 3, pp. 2129-2139, 2015.

[7] G. Jia, G. Han, J. Jiang, N. Sun, K. Wang. Dynamic Resource Parti tioning for Heterogeneous Multi core-based Cloud Computing in Smart Cities. IEEE ACCESS. Vol.4, pp. 108-118, 2016.

[8] G. Jia, X. Li, J. Wan, L. Shi, C. Wang. Coordinate Page Allocation and Thread Group for Improving Main Memory Power Efficiency. In Hotpower conjunction with SOSP'13, 2013.

[9] Lee S,Bahn H,Noh S H. Characterizing memory writere ferences for efficient management of hybrid PCM and DRAM memory [C]//2011IEEE 19th International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS). Singapore: IEEE, 2011: 168-175.

[10] Seok H, Park Y, Park K W, et al. Efficient page caching algorithm with prediction and migration for a hybrid main memory[J]. ACM SIGAPP Applied Computing Review, 2011, 11(4): 38-48.

[11] Seok H, Park Y, Park K H. Migration based page caching algorithm for a hybrid main memory of DRAM and PRAM[C]//Proceedings of the 2011 ACM Symposium on Applied Computing. ACM, 2011: 595-599.

[12] 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.

[13] Wu Z, Jin P, Yang C, et al. APP-LRU: A New Page Replacement Method for PCM/DRAM-Based Hybrid Memory Systems[M]//Network and Parallel Computing. Springer Berlin Heidelberg, 2014: 84-95.

[14] Ramos L E, Gorbatov E, Bianchini R. Page placement in hybrid memory systems[C]//Proceedings of the international conference on Supercomputing. ACM, 2011: 85-95.

[15] G. Han, L. Wan, L. Shu, N. 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

[16] Hassan A, Vandierendonck H, Nikolopoulos D S. Energy-Efficient In-Memory Data Stores on Hybrid Memory Hierarchies[C]//Proceedings of the 11th International Workshop on Data Management on New Hardware. ACM, 2015: 1.

[17] G. Han, W. Que, G. Jia, L. Shu. An Efficient Virtual Machine Consolidation Scheme for Multimedia Cloud Computing, Sensors, Vol.16, No.2, Article 246, 2016

[18] 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.

[19] 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