Semantic-Aware Hot Data Selection Policy for Flash File System in Android-based Smartphones

Dongsoo Choi
Sungkyunkwan University
College of Information and Communication Engineering
Suwon, Republic of Korea
echosoul@skku.edu

Dongkun Shin
Sungkyunkwan University
College of Information and Communication Engineering
Suwon, Republic of Korea
dongkun@skku.edu

***Abstract*—Flash memory has different characteristics from traditional hard disk drives. Therefore, the traditional file systems such as EXT4 are not well-optimized for flash memory storage. Recently, a flash memory-aware file system, called F2FS, is announced, which is based on the log-structured file system considering the poor random write performance of flash memory. Although F2FS uses a heuristic for separating hot and cold data, the heuristic is not aware of file type. In this paper, we observe the lifetime of different types of files in Android platform and propose a semantic-aware hot data selection policy for F2FS. Experimental results show that the proposed technique reduces the garbage collection cost by up to 31%.**

Keywords—Flash Memory; Log-Structured File System; Hot-Cold Data Separation; Garbage Collection; Android System; Mobile Platform

I. INTRODUCTION

Flash memory has many advantages, such as non-volatility, low power consumption, high mobility and high shock resistance. For these reasons, flash memory has been widely used by various mobile devices, such as MP3 player, PDA, smartphone and digital camera. However, the flash memory has a special feature called "erase-before-write" constraint. To update the data of a flash memory page, the corresponding block should be erased before. A flash memory block is composed of several pages. Therefore, flash memory generally uses an out-of-place write mechanism. A special software, called FTL (flash translation layer), is embedded within managed NAND flash devices such as SSD or eMMC, and translates a logical address from host into a physical address. By the address translation, FTL provides a standard block interface. The traditional file systems such as EXT4, which is designed for hard disk drives, can be used for the managed NAND flash devices with the help of FTL.

However, the traditional file systems have no consideration on the characteristics of flash memory. To solve the problem, a new file system targeting for flash memory is recently developed, called F2FS (flash-friendly file system) [1]. Considering the poor random write performance of flash memory, F2FS is designed as a log-structured file system like LFS [2], which generates only sequential writes by copy-on-write mechanism except for meta-data writes. A critical performance bottleneck of log-structured file systems is the garbage collection (GC), where all valid blocks in a victim segment are copied into a free segment. Generally, GC chooses the segment with the smallest valid blocks as a victim segment to minimize the GC cost. To optimize the GC cost further, F2FS writes the hot, warm, and cold data into different segments. Since the data in hot segment may be frequently updated, the corresponding block is invalidated within a short period. Then, the hot segment has many invalid blocks when it is selected as a victim for garbage collection. The current hot/warm/cold separation policy of F2FS determines the directory entry and the file meta-data as hot or warm data. However, the policy has much room for further improvement. In particular, it does not consider the hotness and coldness of regular files. In this paper, we focus on the hot/cold data separation technique for Android-based smartphone. By observing the lifetime of different types of files in Android platform, we design a more enhanced hot data selection policy. With the new policy, we reduced the garbage collection cost by 15% on average in a real Android smartphone device.

II. BACKGROUND

F2FS divided the storage space into blocks. All blocks are 4K in size. Blocks are collected into segments. A segment is 512 blocks or 2MB in size. Segments are collected into sections. F2FS has six sections for writing different sorts of data being written to each one. File content (data) are separated from file meta-data (nodes), and those are divided into "hot", "warm", and "cold" according to types. For example, directory entry is treated as hot and kept separated from file data because they have different life expectancies. Regular file data is expected to remain unchanged for quite a long time, and thus they are treated as cold. The cold section is hardly invalidated, and thus it is not likely to be selected as a GC victim. Nodes are expected to be updated frequently, so a section that was full of hot nodes will have very few blocks that are still live and thus will be cheap to clean.

Although F2FS has a nice hot/cold separation heuristic based on data type, it does not take into account the different lifetime of different file types. F2FS regards all regular file data as cold data and change them as warm data if the file is updated. However, this simple heuristic is not aware of file type. Since there is a correlation between the lifetime of a regular file and its file type, we can predict the hotness or coldness based on the file type. If we collect the hot files into the hot section more aggressively, and thus prevent hot data from sharing a segment with cold data, the garbage collection cost can be further reduced.

III. SEMANTIC-AWARE HOT-COLD DATA SEPARATION

To identify the hot files and cold files in Android systems, we analyzed the write pattern of all files in android-based smartphone while executing several benchmark applications, Web Browser, Google Maps, Youtube and Camera. We measured the update rates of several android system files such as cache files, SQLite DB files, SQLite DB journal files, and XML files. The cache file is an image or media data that is saved temporarily in /cache directory by application. DB journal files are created temporarily for updating database. There are two types of DB journal files, db-journal and db-wal. The db-journal is used for rollback operation whereas the db-wal file is used for write-ahead logging. XML files are used for setting the environment data of applications.

The update rates of cache files in Web Browser, Google Maps, Youtube, and Camera are 68.7%, 6.5%, 2.7%, and 100%, respectively. The update rates are quite different depending on application. XML files also have different update rates. Therefore, the cache files and XML file cannot be determined as hot or cold. SQLite DB file are frequently updated with the update rates of 83.2%, 59.7%, 93.2%, and 26.7%. However, only a small portion of DB file is updated. Therefore, it is not good choice to write all the DB file data at hot segment. The update data of DB file will be treated as warm data by the default policy of F2FS. DB journal files are frequently updated and most of journal data are updated. From this observation, we determined the DB journal files as hot data. When an application creates a DB journal file, the data blocks are written at hot segment.

IV. EXPERIMENTS

We modified the hot data selection policy of F2FS to evaluate the proposed policy. We compared the garbage collection costs of the original F2FS and the modified F2FS on an embedded board. F2FS version 1.0 and Android version 4.0.3 are used. The target storage device is Micro SD Card.

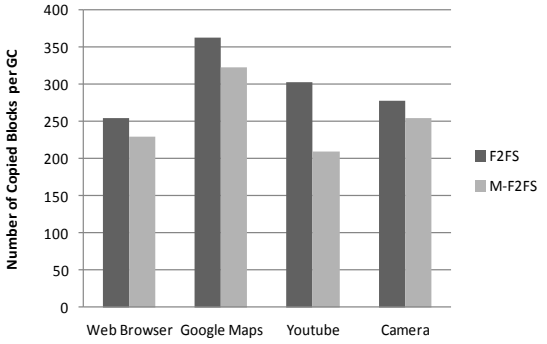

Fig. 1. The comparison on GC costs under different workloads.

We used four real workloads, Web Browser, Google Maps, Youtube, and Camera. With the monkey runner tool, the same input events are generated for the two F2FS implementations. The GC victim selection policy is the Cost-Benefit algorithm.

Fig.1 shows the average number of block copies per GC under different workloads. Compared to the original F2FS, the modified F2FS has lower GC costs.

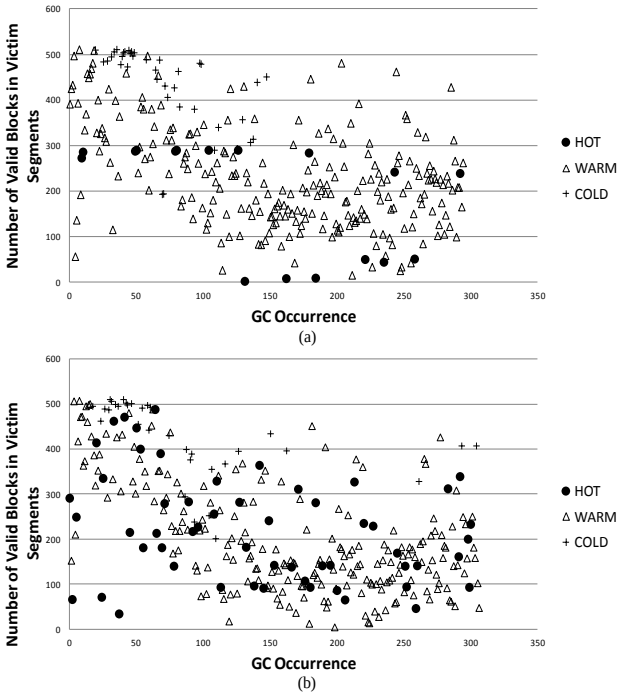

Fig. 2. The number of valid data blocks in victim segments. (a) Original F2FS. (b) Modified F2FS.

Fig.2 shows the numbers of block copies for all GCs during the Web Browser workload. The types of selected victim segments are also shown in the graph. The modified F2FS selects more hot segments as victim compared with the original F2FS. From these results, we can know that the semantic-aware hot data selection policy can significantly reduce the garbage collection cost.

V. CONCLUSION

We investigated the update pattern of several Android platform system files and proposed the semantic-aware hot/cold selection policy for F2FS. The proposed technique significantly reduced the garbage collection cost of F2FS. As a future work, we plan to study the hotness of user files to add more heuristics on F2FS.

ACKNOWLEDGEMENT

This work was supported by the IT R&D program of MKE/KEIT. [KI0018-10041244, SmartTV 2.0 Software Platform]

REFERENCES

[1] An f2fs teardown, http://lwn.net/Articles/518988

[2] Mendel Rosenblum and John K. Ousterhout, "The design and implementation of a log-structured file system," ACM Transactions on Computer Systems (TOCS), Volume 10, Issue 1, Feb. 1992