AN EFFECTIVE SCHEME TO MANAGE STORAGE IN SMARTPHONE DEVICE BY REMOVING DUPLICATE FILES

Ammar Asaad, Ali A.Yassin

Computer Department, University of Basrah, Education College for Pure Sciences, Iraq.

Abstract: During the last decade, mobile usage has increased dramatically. Due to the potential of mobile devices, technology is now widely used by many people. The Internet with the mobile, gives the mobile many additional features like file sharing (images, videos, audios, documents) by social media network apps, exchanging information among relatives and friends. This also means that mobiles receive numerous similar files via applications. As a result, there are many drawbacks, such as save duplicate files in the device's storage and low performance in (CPU, RAM, and battery). In this paper, we suggest and design an improved scheme to eliminate duplicate files. The duplicate files are eliminated using a Message Digest 5 (MD5) hash function, the American Standard Code for Information Interchange (ASCII), and Fast Fibonacci. These methods are used to generate a unique code that can remove the duplicate files in a fast and efficient way. As a result, we got a good result in storage and performance of the mobile device.

Key-words: Duplicate files; Mobile Device; Hash Code, MD5, Fast Fibonacci.

1. Introduction

Simple mobile phones can be utilised for simple computations, while smart devices are used for more complex computations [1]. Mobile devices play a key role in users' daily lives. Mobile devices are overtaking the place of laptops and PCs in all areas of life, including peoples' daily work, communications, and bank transactions work, communications, and bank transformation [2]. In view of social media networks and their high computational complexity, there are many challenges for users in terms of storage space, performance, and energy consumption [1, 2]. Smart device hardware has the ability to perform massive computing tasks, such as image processing, gaming, checking email, bank transfers, transaction authorisations, and other operations [1]. Many people cannot imagine their daily lives without a mobile phone device, or without the internet. Despite these properties, mobile devices still have drawbacks, such as being unable to handle complex tasks like face recognition and games with high-definition graphics[1, 3]. The amount of digitally stored data is growing at a high rate of approximately ten times per five years, overtaking the storage space available. With the increasing use of smartphones such as mobile phone devices and the improvement and development in this field, there is still a low storage problem with the mobile devices [4, 5]. The usage of these devices for a long time causes them to lose storage space, the performance of the CPU decreases, and energy is consumed [6]. One of the solutions to the storage problem is removing duplication files from the mobile phone devices [7].There are many fields based on deleting duplication files, such as paging, image matching, pattern recognition, and similarity of files. Sung-hun et al. [7] suggested a system implemented on a mobile device that is used like mobile apps to reduce the CPU overhead due to the memory deduplication based on the contents of each memory page. Miller et al. [8] proposed a strategy called Cross-layer I/O-based Hints to extend the memory deduplication scanners and then overcome the deduplication overhead rapidly. Furthermore, many authors focused on the main memory of mobile phones, which is one of the most important resources. Byeoksan et al. [9] developed a system known as MemScope that identifies which memory segment contains duplicate memory pages by examining the page table and the memory content. Nohhyun et al. [10] provided a framework to remove duplication files based on dataset contents.

In the field of cloud computing, Waraporn et al. [11] presented an effective data deduplication system for cloud storage to manage storage efficiency and to enhance the performance in the cloud storage environment. Their system computes hash code for each file in cloud storage by applying one-way. The hash functions are implemented in many fields, for example, file coding, authentication, and security. Benjamin et al. [12] described three techniques (summary vector, stream-informed segment layout, and locality preserved caching) to build a fingerprint for each file based on the SHA-1 hash function. They employed these techniques in the deduplication files to remove bottlenecks in the storage.

Currently, with the rapid development in cloud computing and mobile devices, there is a platform to support low latency network access and extra storage called mobile cloud computing (MCC) [13]. Moreover, MCC can provide mobile devices with extra storage based on pay-as-you-go within the cloud computing principle, but customers have to pay for this technique [14, 15]. The duplicate files remain in the storage of the cloud service provider.

Marques et al. [5] suggested a good scheme to delete duplicate files depending on their type. The method works well with known files and suffers to deal with new files that do not exist in the index file of their work [13]. Haustein et al.'s patent [16] involves selecting the deduplication method depending on the file type and computing the deduplication rate, and the deletion of duplicate files is done by the server side. Ryan et al. [13] presented a scheme to reduce energy consumption and the amount of data by detecting duplicate files. However, this scheme initially suffered from low duplicate detection performance and deduplication throughput for a few files. Although their work focused on using cloud storage to deduplicate files, the duplicate files are still in devices' storage. Several researchers focused on detection of duplicate data by using hash functions such as MD5 and SHA-256 [12, 17]. These schemes cannot face the collisions growing the length of the signature increase and the processing time increased with the size of signatures [18]. Ammar et al. [19] proposed a good scheme to delete duplicate images depending on MD5 hash function and Huffman code to generate unique code to remove duplicate image. In this paper, we present a scheme that has been