Journal of Computer Science

Original Research Paper

Novel Scheme for Enhancing Storage Management and Performance and Saving Energy of Mobile Cloud Computing

[1]Ali A. Yassin, [1]Abdulla J. Yassin, [2]Abdullah Mohammed Rashid and [2]Ahmed A. Alkadhmawee

[1]*Department of Computer, Education College for Pure Sciences, University of Basrah, Basrah, 61004, Iraq*
[2]*Department of Computer Science, Education College for Human Science, University of Basrah, 61004, Iraq*

Article history
Received: 14-01-2019
Revised: 16-03-2019
Accepted: 07-05-2019

Corresponding Author:
Ali A. Yassin
Depertment of Computer, Education College for Pure Sciences, University of Basrah, Basrah, 61004, Iraq
Email: ali.yassin@uobasrah.edu.iq
aliadel79yassin@gmail.com

Abstract: Modern industries for smart mobile devices, such as smartphones, provide end users with interactive features that consider network accessibility, computation performance, storage usage, energy saving and limited battery resources. The combination of cloud and mobile computing capabilities has introduced a new technology called Mobile Cloud Computing (MCC), which has extended smartphones and modern mobile devices beyond their core capacities. Many studies have focused only on offloading virtualised applications whilst granting limited consideration on offloading original codes. Moreover, researchers have ignored the main requirements of multimedia, such as photos and videos, in applying the MCC framework in many applications. In this study, we propose a new scheme in the MCC environment to remove duplicated photos received by the mobile device through social network applications. These repeated photos cause loss of large volumes of mobile memory or cloud storage, which leads to difficulty in communication, loss of flexibility to handle more than one application, slow mobile completion and mobile response to orders from another devices and great power consumption due to slow performance. The proposed scheme can overcome the above-mentioned issues, increase the performance of the mobile device and operate at the rest time of the device without affecting the efficiency of its performance. The proposed architecture is based on the management storage from mobile devices to the cloud and uses an efficient search index file to control the addition and deletion operations for repeated photos. Our experiments show that the proposed scheme increases the performance of a certain mobile's application from 1,967,708 ns to 1,708,250 ns, saves energy consumption from 68% to 74% and provides storage space from 10,130 MB to 9,130 MB.

Keywords: Cloud Computing, Mobile Cloud Computing, Mobile Device, Energy Saving, Performance

Introduction

Mobile devices are commonly used in people's daily lives (Hurbungs *et al*., 2016; Lundquist *et al*., 2014). They are called smart mobile devices because they can implement many important tasks simultaneously. Despite the rapid progress in the field of smart mobile technology, mobile resources are more limited than the non-mobile counterparts because of limitations associated with weight, touch capability, storage capacity, battery life and heat dissipation. Limited battery capacity and storage also cannot meet the requirements of current complex applications. Therefore, power-saving techniques, performance computing and management storage are actively being studied to expand the usage time of smartphones. High speed that fully consumes device memory is still the major cause of nerve damage for smartphone users. Although many storage management methods have been proposed, customers are not satisfied with their phone's storage. The power consumption problem of mobile devices indicates the same degree of suffering to the user. When the user interacts with the smartphone, the current system assumes that the user is interactive and must keep the device in active mode. Nevertheless, when the user enters the interaction mode, the device processes the request and displays the result on the device and all the pre-orders if the device is working for some time without connecting to the Internet. During this time, the user cannot obtain any information for understanding and analysing using the phone (Roy *et al*., 2011; Korhonen, 2011). Current technologies are developing the performance of smart mobile devices in terms of storage capability, battery life and speed to suit the requirements of users. However, the processing power of these devices still cannot reach the performance of computers. Limited battery life and computing resources, which depend on the number of applications completed and the remaining space of the storage, are the main challenges. The overall performance of smart mobile devices also depends on the memory, that is, the performance is low when the memory is fully consumed (Danova, 2015; Lewis *et al*., 2014). Mobile Cloud Computing (MCC) tries to solve some of these issues. MCC bridges the gap amongst the limited storage and computing resources of smart mobile devices and the processing requirements of exhaustive applications on these devices (Akherfi *et al*., 2016). The most common mobile cloud architecture is connected with the remote cloud (a sturdy server and software/application), which presents the services to the mobile device over communication channels, such as WAN connection.

Under this structure, mobile devices can access cloud services via the access points or cellular network (Verbelen *et al*., 2012). At present, foremost research topics in MCC are the limited power consumption of mobile devices and the enhancement in users' knowledge through improved lateral application and users' ability in making precise offloading decisions based on multiple elements. Unfortunately, in most states, the size of storage space is increased without considering the memory of the devices and resource utilisation on the cloud service. The main cause in the increase in the size of the mobile space is the receipt of multimedia files of all formats, such as photos, textual data and videos, through social network applications. Social networking has developed considerably in recent years and becomes an important engine for obtaining and disseminating information in different fields, such as science, business, bioinformatics, politics, crisis management and forensics (Verbelen *et al*., 2012; Schober *et al*., 2016). The high prevalence and use of social media is due to the opportunity to send/receive and exchange public messages with low cost anywhere and anytime. Social media platforms accumulate various types of data, including photos, geolocations, videos and audios (Saha and Srivastava, 2014; Shah *et al*., 2015). Therefore, the mobile device receives many files with different formats through social network applications, such as Facebook, WhatsApp and Twitter. Over time, these accumulated data consume a large amount of space allocated to the Mobile Memory (MM) and Cloud Storage (CS). These data fully consume storage space despite the use of an MCC technique, thereby requiring extra cost based on cloud's feature called 'pay as you go.' As a result, accumulated data slow down the computing speed of the mobile device and its capability to operate more than one application simultaneously. Most data associated with multimedia, such as photos and video clips, affect the storage unit. Same photos can be received through a group of users of the same social network or through other applications. The same device can save these same data in many places of storage.

In this study, we aim to solve the aforementioned problems on duplicated photos received by the mobile device through different social networks, which consume the storage space of the device and thus adversely affect its performance. We present a new scheme in the MCC environment to remove such duplicated photos.

The proposed work generates a Search Index File (SIF) on the basis of creating a unique code for each photo by using a crypto-hash function. All photos are grouped by code and only one photo is retained for each group by deleting duplicated photos. The proposed scheme selects the sleeping time of the mobile device depending on its owner's behavior and can thus be applied daily in a smart manner. A Temporary Table (TT) is generated when any photo is added to or removed from the mobile device. During sleeping time, each photo in the TT is checked in the SIF. If it exists in there, then it is deleted from the MM. Otherwise, it is added to the SIF.

Our work has the following essential contributions. Firstly, we identify the sleeping time of the device to perform the owner's request whilst the device is resting. This process aims to remove duplicated photos without affecting the performance of the mobile device. Secondly, we propose the output-oriented power-saving mode of the mobile device. Thirdly, we reduce the processing time of the mobile device by removing unnecessary photos. Fourthly, we create a good foundation to remove unnecessary photos received by the mobile device via social media networks. Fifthly, we implement the sleeping time mode with the Android system that allows to running the proposed scheme without effecting on mobile's system. Sixthly, we describe the challenges in MCC storage. Seventhly, we theoretically analyses the performance, storage management, flexible and efficient search and accuracy of our scheme and conduct extensive experiments on real-world data for validation. Our work focuses on important parts in computation offloading of an MCC organization, such as mobile specifications (available memory and CS, CPU speed and battery level) and application specifications (execution time).

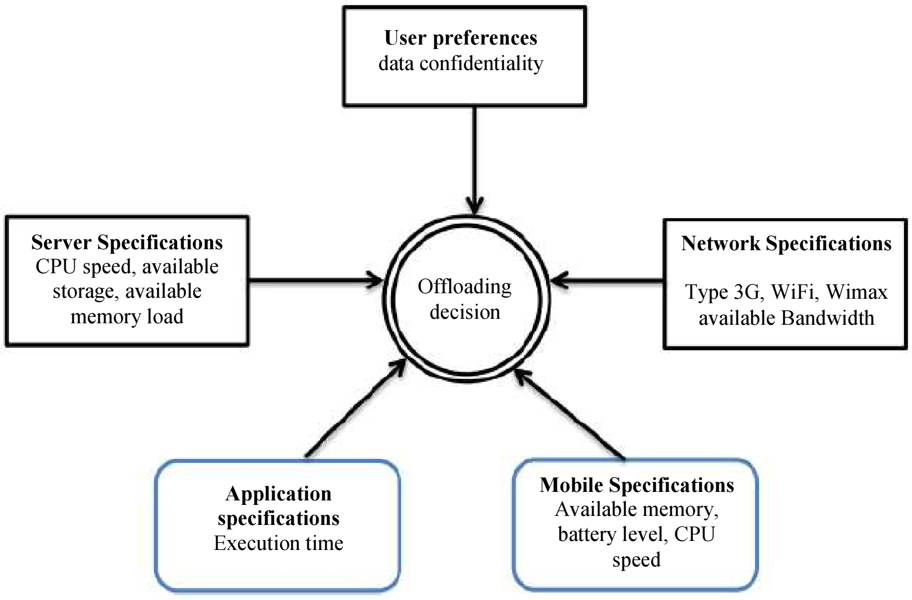

User preferences
data confidentiality
Server Specifications
CPU speed, available storage, available memory load
Offloading decision
Network Specifications
Type 3G, WiFi, Wimax available Bandwidth
Application specifications
Execution time
Mobile Specifications
Available memory, battery level, CPU speed

Fig. 1: Main parts affecting the offloading decision in the mobile environment

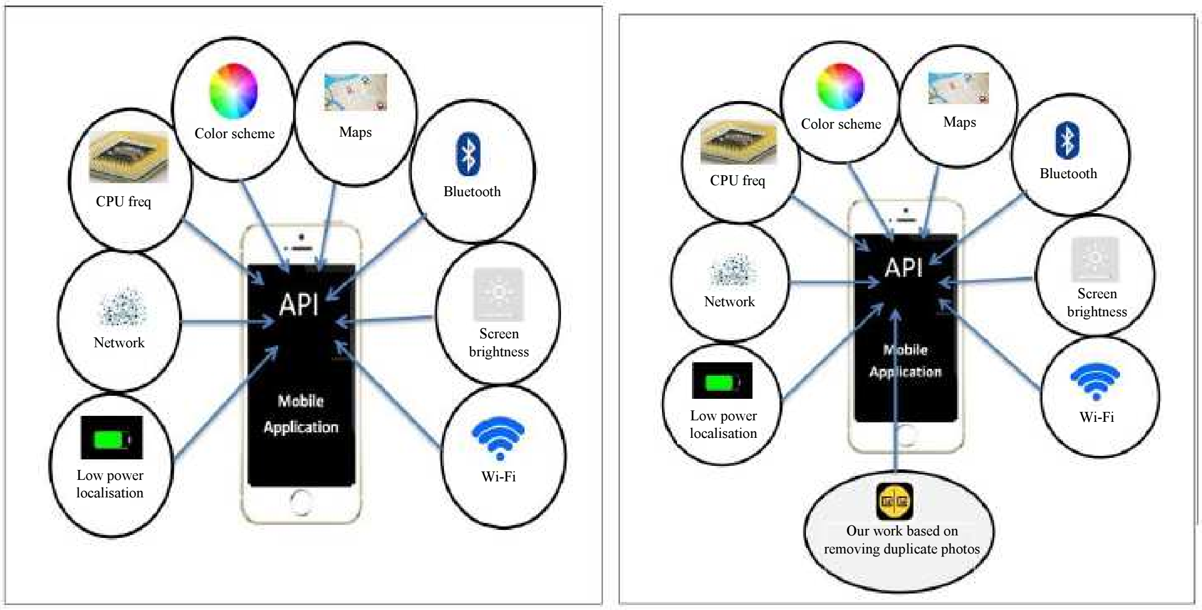


Fig. 2: Contributions in power-saving API diagram

Figure 1 shows the main parts affecting the offloading decision in the mobile environment. Android applications are important in maintaining and saving energy in mobile devices. Figure 2 demonstrates our contributions in the diagram of power-saving API. Nearly full memory affects the CPU performance in executing any application and the power of the mobile device. The memory becomes full when the mobile device receives duplicated photos from various social networks.

The rest of the paper is structured as follows. The next section reviews related works on performance, storage management and power saving in mobile devices. Next, we describe the proposed scheme and develop suggestions. The research methodology is then described, followed by a discussion of the key findings in performance, power saving and storage management. The experimental results are presented with the data analysis. The last section shows the contributions of this study.

Related Works

The commonly used scheduling in mobile devices and MCC environment aims to increase performance and save energy. In the information technology era, many studies support remote implementation for mobile phone applications on the cloud to increase performance and reduce power consumption (Shen *et al.*, 2016; Shiraz *et al.*, 2013). The customisable task scheduler is suggested in (Lu *et al.*, 2005) to manage tasks between the mobile device and CS. It is customisable because the user can arrange his or her main goals (such as power consumption and performance) depending on the function in scheduling tasks and determining their importance. Magurawalage *et al.* (2014) proposed a scheme based on two points: (1) Delay task, which is consists of execution time and queuing time at the processing device in the network with communication time; and (2) power saving, which is composed of computation power consumed at the processing device and communication energy along the path. This scheme provides an accurate offloading code to achieve the highest energy. It uses dynamic lateral application and partitioning policy for offloading. As a result, this lateral process causes an overhead on the smartphone.

Sprecher (2016) proposed an algorithm to select whether the application will execute internally or remotely using the cloud. This algorithm considers the mobile application as a single component that cannot be divided into multiple approaches. However, some approaches should be executed internally and others should be offloaded to the cloud service provider.

Shiraz *et al.* (2015) presented a flexible framework for detecting offloading processes. This framework uses services based on the remote server nodes and virtual devices to implement offloading of mobile applications. It needs synchronisation between the mobile and virtual devices in the first round and synchronisation between the virtual device and server node in the second round, thereby requiring many resources.

Most parts of everyday life can be influenced by the usage of smartphones (McDaniel and Coyne, 2016; Misra *et al.*, 2016). Smartphones can impact individuals' relational life (Mok *et al.*, 2014; Beloglazov *et al.*, 2012) and the redundant use of smartphones can lead to intemperance and decreased capacity to enjoy comfort. The exchange of dialogues, pictures and videos can be useful and harmful at the same time. Many of these exchanged messages will be repeated, occupy areas in memory and even go beyond the capacity of the CS (Beloglazov *et al.*, 2012).

As mentioned earlier, our work focuses on eliminating repeated photos that occupy a large space of the MM and CS. We design a smart and fast way to build an index file for all photos on the device by building a code for each image. A mathematical equation is suggested to build a unique code for each photo. The photos are then categorised using the codes to group similar images. All identical photos are then erased, such that one remains in the MM and another backup is stored in the CS. Next, the proposed scheme detects the sleeping time of the smartphone. Any photo supplied to the device via social networks will then be assigned its own code and placed in a TT. After the sleeping time is determined, each photo in the temporary file is searched within the indexed file. If it is found in there, then it is removed from the MM; otherwise, it is added to the index file and sent to the cloud service provider. Experimental results indicate that the proposed work manages the reservoir area in an intelligent manner by removing redundant data, increasing the performance of any application's response and reducing energy consumption.

Proposed Scheme

Our proposed scheme is composed of four phases: Initialisation, building effective SIFs, provision of storage and sleeping time. The first phase prepares and detects all photos inside the MM and CS, whereas the second phase generates the main SIF. The index file is generated in two parts: The first part is produced using the photos stored inside the memory of the mobile device (such as smartphone), whereas the second part is related to the CS provider. The provision of storage phase removes all the repeated photos in different folders within the MM. The main work of this phase depends on the index file of the MM. The fourth phase focuses on rebuilding an effective SIF that is related to the MM. Thus, the processing time of this phase uses a sleeping time algorithm that detects the appropriate time for running this phase without affecting the operation of the smartphone. The algorithm works smartly depending on the sleeping time of the smartphone.

The fourth also phase rearranges the index file again when the smartphone receives new images or when the mobile owner deletes images from the smartphone. The symbols used in our proposed scheme are shown in Table 1. Figure 3 shows the architecture of our proposed scheme.

Initialisation Phase

Mobile's Data Owner (MDO) has a collection of photos distributed in different folders in MM, such as Gallery, Facebook and WhatsApp. Let S = {Pho_1,…, Pho_n} refer to the set of *n* photos in MDO's collection in different places inside MM. S includes a set of photos that are outsourced from the MM to CS. In Samsung mobile devices, the outsourcing folder that saves photos in the CS is called Auto Backup. Google offers the same service known as Photos service inside Google Drive, which is used by many mobile companies, such as Samsung and Huawei. In this phase, our objective is to scan all photos saved in MM and CS and then detect the path and size of each photo. The main steps of this phase are summarised as follows:

- Scan and accumulate all O's photos in S = $\{Pho_1,\ldots,Pho_n\}$
- Let P = $\{P_1,\ldots, P_1\}$ denote the set of photos' paths

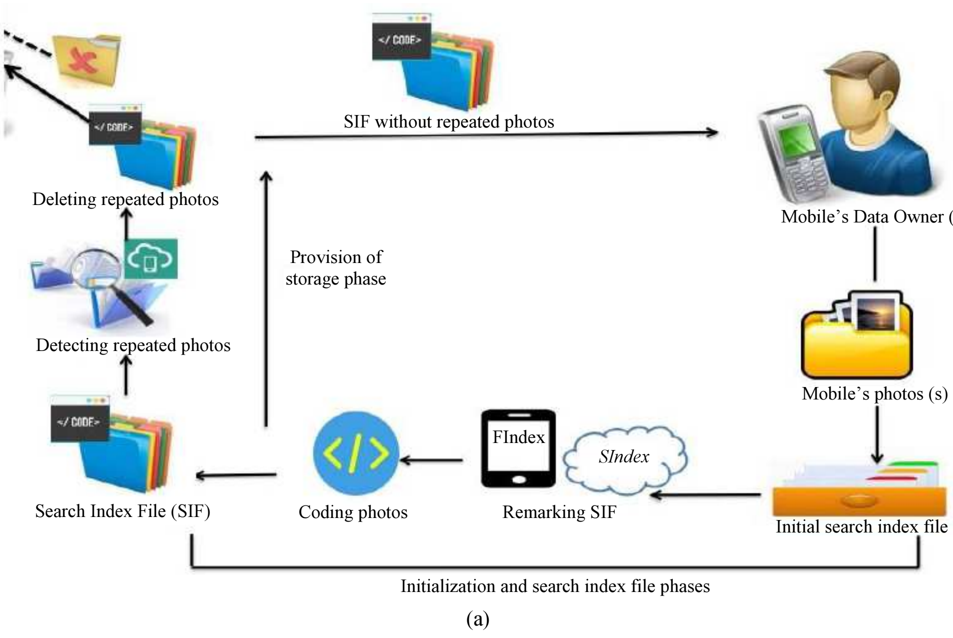
Compute the total size of MDO's photos by using Equation 1 and then determine the ratio of storage space occupied by the photos from the MM by using Equation 2:

$$T = \sum_{i=1}^{n} Size\left(Pho_i\right) \tag{1}$$

$$RS = \frac{\left(Size\left(MM + CS\right) - T\right)}{Size\left(MM + CS\right)} \times 100 \tag{2}$$

where, *Size*(*CS*) represents the storage space that is reserved by MDO from the cloud service provider.

If RS>50%, then output 'Warning Message'.


SIF without repeated photos
Deleting repeated photos
Mobile's Data Owner (MDO)
Provision of storage phase
Detecting repeated photos
Mobile's photos (s)
FIndex
SIndex
Search Index File (SIF)
Coding photos
Remarking SIF
Initial search index file
Initialization and search index file phases
(a)

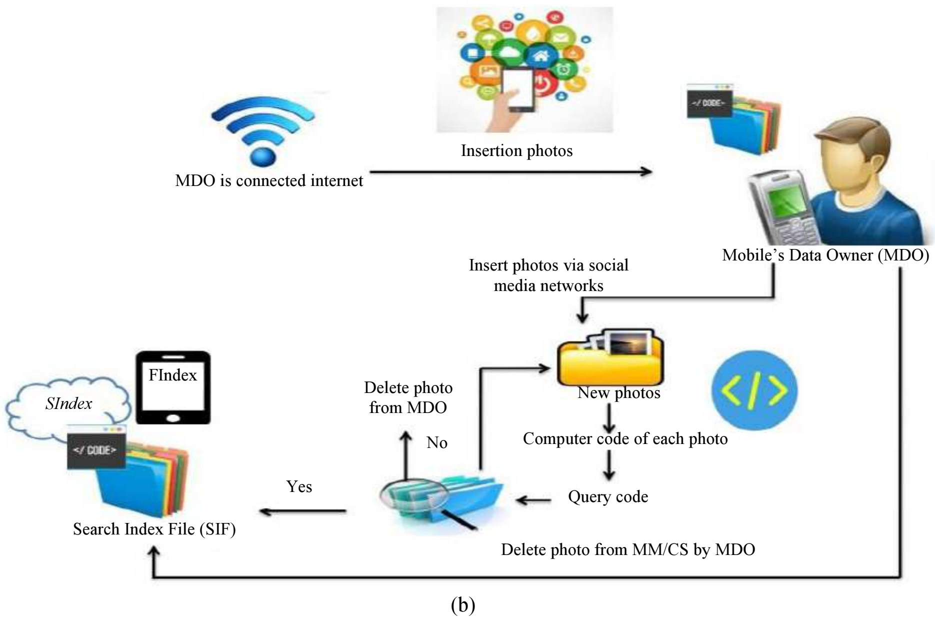

Insertion photos
MDO is connected internet
Mobile's Data Owner (MDO)
Insert photos via social media networks
FIndex
SIndex
Delete photo from MDO
New photos
No
Computer code of each photo
Yes
Query code
Search Index File (SIF)
Delete photo from MM/CS by MDO
(b)

Fig. 3: Mechanism of our proposed scheme; (a) Initialisation and Search Index File Phases; (b) Provision of Storage and Sleeping Time Phases

Table 1: Description of symbols used throughout our proposed scheme

Symbol	Meaning
MDO	Mobile's data owner: owner of mobile devices, such as smartphones and tablets.
MM	Mobile memory
CS	Cloud storage with Auto Backup of MDO
Pho	Photo inside MDO's device
S	Set of mobile photos saved in MM and CS
P	Set of paths of mobile photos
Size()	Function for computing the size of each photo
T	Total sizes of MDO's photos
RS	Ratio of T to size of MM and CS
SIF	Search index file
Flndex	Part of SIF inside MM
Slndex	Part of SIF inside CS
Code_i	Unique code for each photo
h_i	Result of applying MD5 on Pho_i
Filtering	Function for filtering photos depending on their paths in MM or CS
DetRepeating	Function for detecting and deleting repeated photos whilst keeping one version
TimeO, TimeC	Sleeping time of MDO's device
Curr-Date	Function for returning current date
Total	Mobile usage time
Sort	Function for quickly sorting and returning the most common times (TimeO, TimeC) of the week
Search	Function for searching Code_i in SIF; if the item is found, then return True; otherwise, return False.
DeletingMemory	Function for deleting photos from MM
DeleteRec	Finction for deleting records from SIF

Building Effective Search Index File Phase

The memory and processor of mobile devices are limited. Thus, the process of disposal of repeated photos needs to consider this fact. Therefore, image matching methods that rely on feature extraction of images are unsuitable for the aforementioned task. Image matching methods involve many stages, such as pre-processing, feature extraction and other phases, in achieving the final characteristics depending on the generated index file for finding match ratios. These methods are difficult to apply to mobile devices. Accordingly, the proposed scheme depends on the construction of an index file in a fast and efficient manner. This index file will be used to remove repeated photos in the subsequent phases without affecting the performance of the mobile device.

Two sets (S,P) are obtained from the first phase to build an effective SIF and the file is divided into two parts depending on the photos' paths *P*. The file is divided into two parts due to two reasons. Firstly, Flndex is linked to the MM, whereas Slndex is associated with the cloud service provider, which requires connecting to the Internet when we need to modify the file (SIF) as a result of the insertion or deletion operation for photos from the mobile. Secondly, we store a backup version of each photo in the user's CS provider. This photo will be repeated inside the SIF. The process of removing the redundancy from the file completely will result in the loss of the outsourced photos in the CS. The following steps describe the mechanism of this phase.

Classify mobile photos into two groups depending on the storage type (path set).

```
Algorithm 1 of Classification
Input: $\{P_i\}$, $1 \le i \le n$; where n is number of photos' paths
Output: SIF
        For $I$ = 1 to n do
            If $P_i$ Is Outside of MM Then
                SIF($P_i$, Slndex)
            Else
                SIF($P_i$, Flndex)
            End if
        End for
End Algorithm 1 of Classification
```

Part (Slndex) of SIF is connected to the Internet via the presence of its photos in the CS side. Therefore, Algorithm 1 is applied on the photos stored in MM marked Slndex and then works on the Slndex photos depending on the connection to the Internet.

An effective SIF is built by using Algorithm 2 in the subsequent phases. Algorithm 2 is applied to Equations (3) and (4) to generate a unique code for each mobile photo:

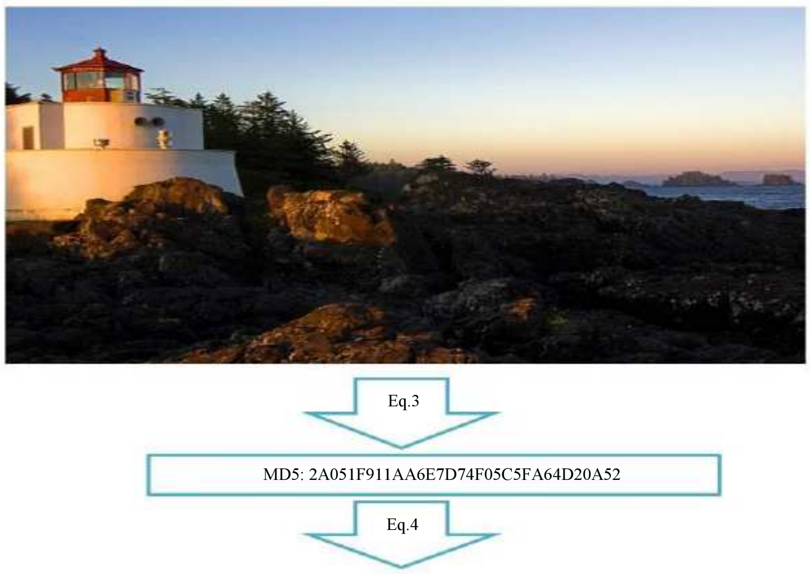
$$h_i = MD5\left(Pho_i\right) \quad (3)$$

$$Cod_i = \sum_{j=1}^{L} ASCII\left(h_i\left(j\right)\right)^{\wedge} Weight\left(h_i\left(j\right)\right) \quad (4)$$

where, L is the length of h_i, *ASCII* is the function for computing *ASCII* code for each symbol inside h_i and *Weight* is the function for assigning each symbol in h_i a numeric serial value depending on its appearance in the symbolic string h_i. Figure 4 shows the working mechanism of Equation 4 for real-world data extracted from a 'Samsung SM-G900H 4d0079e94bb9413f' smartphone.

Algorithm 2 of Search Index File

```
    Input: {$S_i$, $P_i$}, $1 \le i \le n$; where $n$ is number of photos
    Output: SIF
    For $i$ = 1 to n do
        Compute $h_i$ as in Equation (3)
        Compute $Code_i$ as in Equation (4)
        SIF($Code_i$, $P_i$, Size($Pho_i$))
    End for
End Algorithm 2 of Search Index File
```



Wight	1	2	3	4	5	6	7	8	9	10	11	12	13	---------	40
MDS	2	A	0	5	1	F	9	1	1	A	A	6	E	---------	2
ASCII	50	41	48	53	49	46	57	49	49	41	41	54	45	---------	50
Eq.4 (code$_i$)	50^1	41^2	48^3	53^4	49^5	46^6	57^7	49^8	49^9	41^10	41^11	54^12	45^13	---------	50^40
	1686616022														

Fig. 4: Mechanism of computing code for each mobile photo

Index file before applying provision of storage phase

S.	Path of photo	Code of photo
1	/storage/emulated/0/Pictures/Instagram/IMG_20170423_190112_275.jpg	1258744707
2	/storage/emulated/0/Pictures/Instagram/IMG_20170711_191807_273.jpg	469187893
3	/storage/emulated/0/FilmoraGo/Multimedia/img-16752098.png	611023240
4	/storage/emulated/0/DCIM/Camera/56789.jpg	262822616
5	/storage/emulated/0/Pictures/Instagram/IMG_20170711_191807_273.jpg	469187893
6	/storage/emulated/0/Pictures/Instagram/IMG_20170423_190112_275.jpg	1258744707
7	/storage/emulated/0/FilmoraGo/Multimedia/img-16752098.png	611023240
8	/storage/emulated/0/Pictures/Instagram/IMG_20170423_190112_275.'jpg	1258744707
9	/storage/emulated/0/FilmoraGo/Multimedia/img-16752098.png	611023240

Index file after applying provision of storage phase

S.	Path of photo	Code of photo
1	/storage/emulated/0/Pictures/Instagram/IMG_20170423_190112_275.jpg	1258744707
2	/storage/emulated/0/Pictures/Instagram/IMG_20170711_191807_273.jpg	469187893
3	/storage/emulated/0/FilmoraGo/Multimedia/img-16752098.png	611023240
4	/storage/emulated/0/DCIM/Camera/56789.jpg	262822616

Fig. 5: Viewing of provision of storage phase

Provision of Storage Phase

The major challenge in mobile devices is to provide sufficient storage space for applications and multimedia data, such as photos, videos and audios. With the fast increase in required data storage, providing high efficiency for the mobile whilst maintaining and managing the capacity of the memory of the device becomes difficult. The repeated photos saved in the MM or CS fully consume the memory space of the mobile device, which leads to its slowdown. In this phase, we focus on removing repeated images either on the MM or CS. The removal of repeated images depends on the image code computed in the previous phase. We notice that the SIF contains ($Code_i$) and P_i. It will extract and accumulate photos into groups depending on the image code ($Code_i$). Each group contains similar images and are in different paths (P) depending on social media networks' folders. The following steps demonstrate the basic mechanism of this phase:

1. Detect all repeated photos from SIF based on $Code_i$. As a result, the records inside SIF distribute in blocks depending on the value of $Code_i$
2. Delete all repeated photos from SIF whilst saving the version of photo on MM and CS. Algorithm 3 explains the mechanism of removing repeated photos. Figure 5 illustrates the mechanism of this phase based on real-word data obtained from a 'Samsung SM-G900H 4d0079e94bb9413f' smartphone

```
Algorithm 3 Deleting
  Input: {SIF}
  Output{SIF without repeated photos}
        Filtering (SIF Flndex) // is used to filter photo
        depending on its path in MM or CS.
  DetRepeating (SIF) // it is used to detect and delete
  repeated photo.
    If Mobile Is Connected to the Internet Then
            Filtering (SIF, Slndex)
            DetRepeating (SIF)
      End if
        // Statements of CS. Otherwise, these
        statements are run automatically when mobile
        is connected
End Algorithm 3 Deleting
```

```
Algorithm 4 Detect Sleeping Time
  Input: {MDO}
  Output{TimeO, TimeC}
  For $i$ = Curr-Date down to Curr-Date -7 Step -1
     TimeO = MobileIsOpen(MDO) // Mobile unlock
     time.
     TimeC = MobileIsClose(MDO) // Mobile lock
     time.
     Total = TimeC – TimeO // Mobile usage time
     Insert into Table values (TimeO, TimeC, Total, $i$)
  End For
    [TimeO, TimeC]= Sort (Table, Total )
End Algorithm 4 Detect Sleeping Time
```

Sleeping Time Phase

The smartphone receives hundreds of photos through social media network applications every day. Many of these photos are duplicated, thereby affecting the storage spaces of the MM and CS. This phase works smartly as it monitors device usage of the device owner for each week to detect sleeping time. The device usage refers to the times the mobile is turned on and off by the owner and determines the times that the device is left for hours by the owner during his or her bedtime or other works. Here, a table is built on the basis of a statistical procedure (Algorithm 4) on the hours of device usage by the owner to determine the approximate sleeping time of the device. This phase determines whether a new image is added in the mobile device via social media networks and decides whether a photo will be added in the index file. This phase also modifies the index file if the deletion of a single photo or a set of photos is carried out by the owner of the mobile device. The following steps illustrate the mechanism of this phase.

Determine the sleeping time by using Algorithm 4. This algorithm works in an intelligent manner on the basis of the behaviour of the mobile's owner. Assume the owner has left his or her mobile from 12:00 am to 5:00 am. The time execution of the current phase is exclusive during this period. After 10 days, if the owner leaves the mobile device from 3:00 am to 7:00 am, then the execution time changes automatically.

Detect the new photos $S^* = \{Pho_1^*,...,Pho_n^*\}$ that have been added to the mobile device via social media networks depending on the number of days. This step determines the photos that are deleted by the device owner and then keeps the path of all deleted photos $P^* = \{P_1^*,...,P_n^*\}$. Here, this step generates a TT that consists of S^* and P^* (Table 2).

Update SIF depending on TT. Algorithm 5 is responsible for updating SIF. The addition occurs after confirming that a new photo Pho_i^* is not found in SIF based on computing the photo code $Code_i$ by using Equations (3) and (4). Lastly, a new photo will be added to MM and CS. The deletion process is continuously performed depending on P^* to remove all photos from SIF. The proposed work gives an explanatory message to the owner of the device if he or she wishes to remove the same photos from the CS.

```
Algorithm 5 Updating
  Input: {SIF, TT →
    $S^* = \{Pho_1^*,...,Pho_n^*\}$, $P^* = \{P_1^*,...,P_n^*\}$}
```

```
Output{SIF}
For $i$ = 1 To $n$
        Compute $h_i$ as in Equation (3) for $Pho_i^*$ based on $P_i^*$
        Compute (Code$_i$) as in Equation (4)
        If Search (Code$_i$, SIF) == False Then
    SIF(Code$_i$, P$_i$, Size(Pho$_i$), FIndex) // Add a new photo to MM
    SIF(Code$_i$, P$_i$, Size(Pho$_i$), SIndex)// Add a new photo to CS
Else
  DeletingMemory (Pho$_i$) // Delete from MM.
End If
    End For
  For i = 1 To $n$
    DeleteRec (SIF, P$_i$)
End For
End Algorithm 5 Updating
```

Proposed Scheme Evaluation

Analysis and Experimental Results of the Proposed Scheme

We analyse our proposed scheme and demonstrate its properties, such as effectiveness and flexibility. We also calculate the computational and communication costs of our work.

Theorem 1

Our work conducts flexible and efficient search.

Proof

Our proposed scheme constructs the main SIF to accelerate the image search process inside MM and CS. In the initialisation phase, the data owner first executes the proposed scheme to extract a descriptor component for searching {S$_i$, P$_i$} each of which requires the computation of O(n). This phase also computes the percentage of storage space occupied by images in the MM and CS. The device owner receives a warning message in case RS >= 50. SIF consists of each photo's path (P$_i$), size, code (Code$_i$) and type of storage: FIndex for MM's photos and SIndex for CS's photos. SIf has flexibility to work on photos saved in the MM and CS.

SIF works first on images stored in the mobile device and then moves on to those stored in the cloud. If the device is connected to the Internet, then SIF is built completely. Otherwise, it only works on MM's photos and then handles CS's photos directly when the Internet is available. This flexible processing also works the same way when deleting duplicated photos from CS regardless of the availability of the Internet. The deletion is postponed until the availability of the Internet. The initialisation phase works only once when the proposed method starts running at the sleep time of the mobile device. This phase does not affect the effectiveness of the search process in real time. Improvements in the implementation may be made through customising it towards the procedure of SIF updating through adding/deleting photos to the mobile device, which leads to updating SIF partially. Therefore, our proposed scheme has flexibility and efficiency features.

Theorem 2

Our proposed scheme provides a unique code for each photo in the mobile device.

Proof

An image matching method is applied due to numerous practically applicable computational challenges in obtaining the similarity between entities demonstrated in digital images. Therefore, the proposed scheme focuses on building a unique code Code$_i$ for each photo inside the MM. The purpose of Code$_i$ is to decrease the processing time of image matching to suit the limited possibilities of mobile devices. By applying Equations (3) and (4), we obtain Code$_i$ based on hash function MD5, which is adopted to ensure fast processing time and accuracy in retrieving the desired image. Figure 6 shows the processing time of MD5, SHA-256 and SHA-512 on a set of images stored on a 'Samsung SM-G900H 4d0079e94bb9413f' smartphone. In Phase 3, the proposed scheme groups photos through code matching to remove all duplicated photos. When inserting any photo Pho$_i$ to the mobile device, Code$_i$ is first computed and then searched inside SIF to ensure Pho$_i$ exists on the mobile device. If it exists, then it is deleted from MM and CS. Otherwise, it is added to SIF. The same situation is observed in deleting any photo Pho$_i$ from the mobile by MDO. Our work in Phase 4 keeps the code of this photo and then deletes the information of Pho$_i$ in the sleeping time phase based on Code$_i$. Hence, Code$_i$ is important for the efficiency of our proposed scheme.

Theorem 3

Our proposed scheme provides high-precision and high-performance search to any photo.

Proof

The proposed work needs very high- precision search for a photo and should reach 100%. The receipt of a new photo $Phot_i^*$ by the mobile device through social networks generates a TT containing the name of the photo and its path P_i^*. In the sleeping time phase, the search for each photo in TT begins by computing Code$_i$ of $Phot_i^*$ and then determining whether it is present within Sif. If Code$_i$ is found in SIF, then Pho$_i$ is deleted from MM and CS. Otherwise, $Phot_i^*$ and its details will be added to SIF. The high-performance search for each $Phot_i^*$ is due to that Code$_i$ is obtained using Equations (3) and (4), thereby resulting in the error rate in accuracy of up to zero. Table 2 shows the sample of TT from real data obtained from a 'Samsung SM-G900H 4d0079e94bb9413f' smartphone. TT has been created on December 2, 2018 and the processing time for generating TT is 1,378,425,250 ns. Figure 7 shows the accuracy of results for photos in TT. Figure 8 presents the time performance for adding $Phot_i^*$ to SIF or deleting it from MM when $Phot_i^*$ exists in SIF.

Theorem 4

Our proposed scheme provides search linkability.

Proof

In the second phase, the photos are aggregated in groups depending on similarity in codes as a first link step. Consequently, this process is executed once in the first moments of operating the proposed scheme and returns all photos associated with the same code. This step is highly efficient because all the linked images are grouped together with the same code once. Table 3 views the mechanism of linkability feature in the second phase. The same applies to new photos from the mobile device through social networks in the third and fourth stages. The search query for duplicated photos have the same code despite coming from different social networks, thereby guaranteeing the linkability of different search queries.

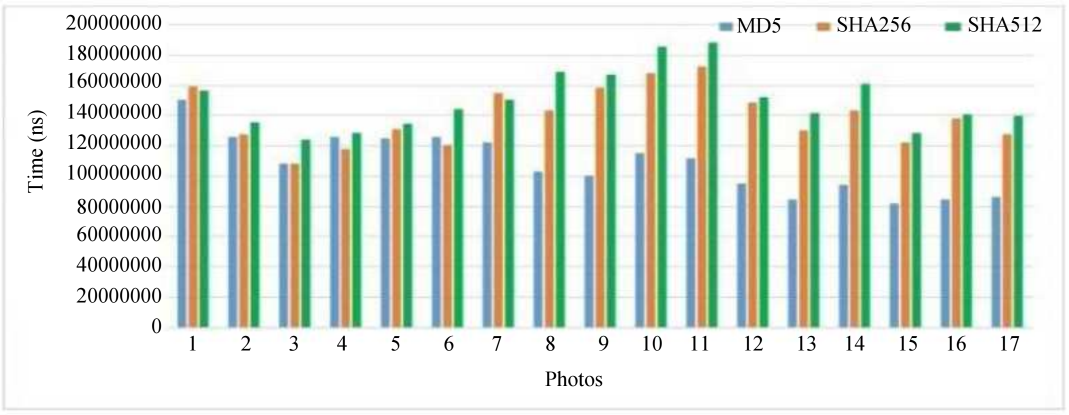


Fig. 6: Performance of hash functions

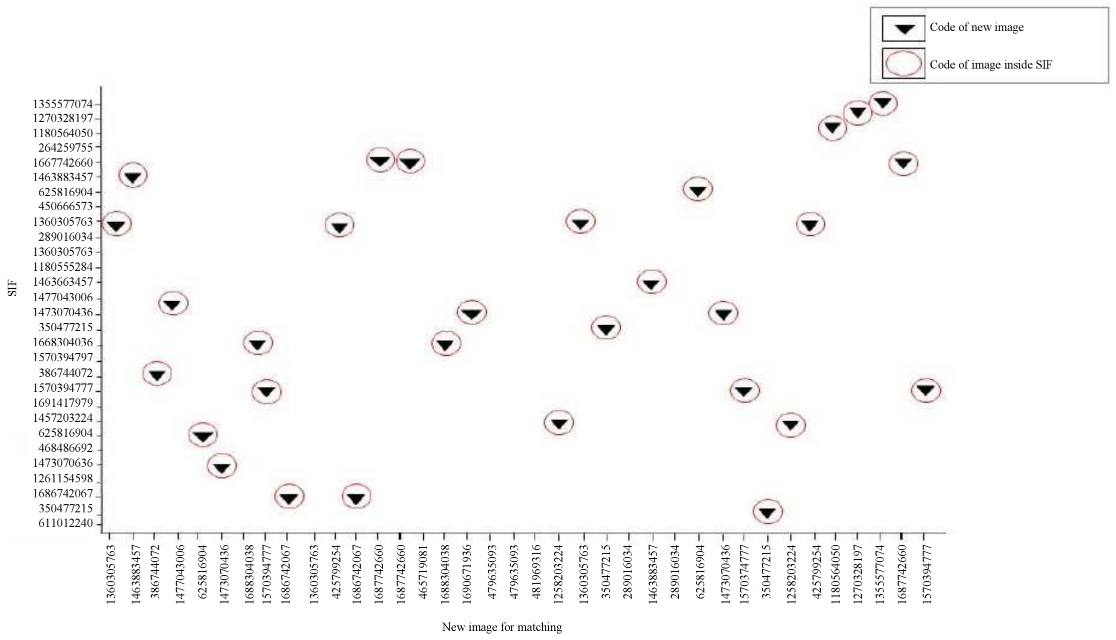


Fig. 7: Accuracy of results by looking for a new photo in the SIF

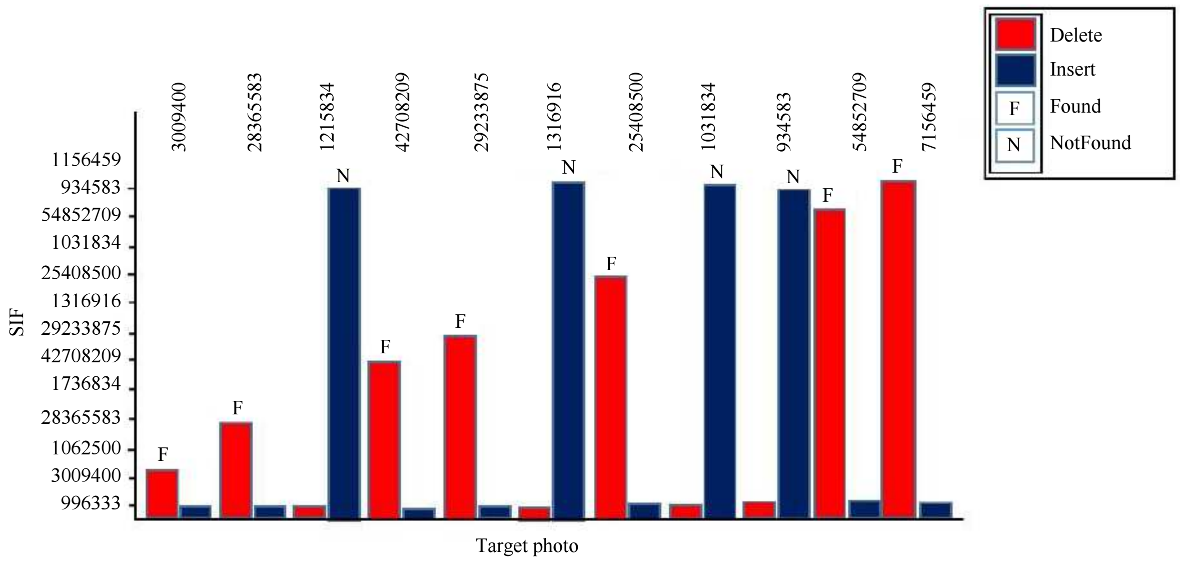


Fig. 8: Working mechanism of insertion or deletion operation for a photo from the device based on the search process. Insert refers to inserting a new photo to SIF based on the result of searching (N), whereas Delete refers to deleting photos from SIF based on the result of searching (F). The result of searching for a new photo is either F or N and leads to performing only one of deletion or insertion operation

Table 2: Sample of TT

S	Photo path	Photo name	Operation type
1	/storage/emulated/0/Pictures/Instagram/IMG_20180203_161987_359.jpg	IMG_20180203_161987_359.jpg	Add
2	/storage/emulated/0/Pictures/Instagram/IMG_201801189_085584-698.jpg	IMG_201801189_085584-698.jpg	Add
3	/storage/emulated/0/WhatsApp/Media/WhatsApp Images/WhatsApp Image 2017-10-19 at 8.23.23 PM.jpeg	WhatsApp Image 2017-10-19 at 8.23.23 PM.jpeg	Deleted
4	/storage/emulated/0/WhatsApp/Media/WhatsApp Images/WhatsApp Image 2017-11-23 at 2.43.16 PM.jpeg	WhatsApp Image 2017-11-23 at 2.43.16 PM.jpeg	Deleted
5	/storage/emulated/0/WhatsApp/Media/WhatsApp Images/WhatsApp Image 2017-12-13at 12.41.11 PM.jpeg	WhatsApp Image 2017-12-13at 12.41.11 PM.jpeg	Add
6	/storage/emulated/0/WhatsApp/Media/WhatsApp Images/WhatsApp Image 2017-12-01 at 12.58.17 PM.jpeg	WhatsApp Image 2017-12-01 at 12.58.17 PM.jpeg	Add
7	/storage/emulated/0/viber/media/Viber Images/IMG-455d6eb0c18c1921f6ce4f8877784480-V.jpg	IMG-455d6eb0c18c1921f6ce4 f8877784480-V.jpg	Add
8	/storage/emulated/0/viber/media/Viber Images/IMG-4ccf6424232478645bf521b6ec9f292a-V.jpg	Viber Images/IMG-4ccf64242324 78645b f521b6ec9f292a-V.jpg	Add
9	/storage/emulated/0/viber/media/Viber Images/IMG-5b761bf501ef00d78b6f95cafb054140-V.jpg	IMG-5b761bf501ef00d78b6 f95cafb054140-V.jpg	Deleted
10	/storage/emulated/0/viber/media/Viber Images/IMG-bb6248a36b4a75sgsrrhdddtbf6750a3-V.jpg	Viber Images/IMG-bb6248a36b4a75 sgsrrhdddtbf6750a3-V.jpg	Add

Theorem 5

Our proposed scheme is maintained on SIF in a smart manner.

Proof

The proposed work monitors the photos added to the mobile device or deletes them intelligently without impacting the performance of the device. In the fourth phase, a TT will be created, including the name and path of each photo and the type of operation performed on it (added or deleted), for the received photos through social networks or the deleted photos by MDO.

Algorithm 4 is used to determine the sleeping time. This algorithm works intelligently depending on the resting time of the mobile device obtained using the behaviour of the mobile's owner. The maintenance of SIF is done by deleting or adding photos from the storage area without impacting the effectiveness and completeness of the device.

Table 3: Repeated photos in different folders on social networks

S	Code	Path
1	1180555284	/storage/emulated/0/viber/media/Viber mages/IMG-455d6eb0c18c1921f6ce4f887778440-V.jpg
		/storage/emulated/0/viber/media/Viber Images/IMG-455d6eb0c18c1921f6ce4f885d6e469-V.jpg
		/storage/emulated/0/viber/media/Viber Images/IMG-5d6eb0c18c134f7hs381f6ce4f887778-V.jpg
		/storage/emulated/0/viber/media/Viber Images/IMG-5d6eb0c18c1sfrwei381f6ce58d76914-V.jpg
2	1258744707	/storage/emulated/0/Pictures/Instagram/IMG_20180118_124578_185.jpg
		/storage/emulated/0/Pictures/Instagram/IMG_20170711_190112_275.jpg
		/storage/emulated/0/Pictures/Instagram/IMG_20180203_161987_2359.jpg
3	1270328197	/storage/emulated/0/viber/media/Viber Images/IMG-c3e1d43c93a0c94835365dc7f38eacd4-V.jpg
		/storage/emulated/0/viber/media/Viber Images/IMG-c3e1d43c9325d85ka0c94835365dc7f3-V.jpg
4	625880964	/storage/emulated/0/Pictures/Instagram/IMG_201712144_297422_966.jpg
		/storage/emulated/0/Pictures/Instagram/IMG_20180203_161987_359.jpg
5	608219957	/storage/emulated/0/WhatsApp/Media/WhatsApp Images/WhatsApp Image 2018-1-20 at 6.5.54 PM.jpeg
		/storage/emulated/0/WhatsApp/Media/WhatsApp Images/WhatsApp Image 2017-11-20 at 10.57.35 PM.jpeg
		/storage/emulated/0/Pictures/Instagram/IMG_20180116_121427_966.jpg
6	1166675612	/storage/emulated/0/WhatsApp/Media/WhatsApp Images/WhatsApp Image 2017-10-01 at 8.58.26 PM.jpeg
		/storage/emulated/0/WhatsApp/Media/WhatsApp Images/WhatsApp Image 2018-01-14 at 9.25.16 PM.jpeg

Theorem 6

Our proposed scheme provides green computing.

Proof

The principle of our proposed scheme supports green computing because it aims to primarily reduce energy consumption, which meets with the features of green computing as an energy-aware characteristic (Gai *et al*., 2016). The methods of accessing green computing are from hardware and software to managing, strategy and authorised issues. Our work focuses on the computing side that uses dynamic programming with mobile cloud computing. Our proposed work can be used to reduce energy consumption in mobile devices. The proposed scheme can also measure the steadfastness of the system, which guarantees its performance. Green computing is performed to enable the use of our proposed scheme to reduce energy consumption in mobile devices without weakening the performance of cloud services. In the sleeping time phase, the maintenance of the MM is run during the rest time of the mobile device. This process aims to remove all duplicated photos from the mobile device, which increases the performance of the mobile and spaces of ram, MM and CS. Algorithms 3, 4, 5 and 6 are used to maintain such storage spaces.

We examine the performance 10 days after applying the proposed work to a 'Samsung SM-G900H 4d0079e94bb9413f' smartphone. The proposed scheme exhibits high performance when receiving new photos via social media applications and then ensures no repeated photos are stored through the SIF. High efficiency is also observed for photos deleted by the owner of the device. Table 4 and 5 show the performance of the mobile device during deletion and insertion operations before and after executing the proposed scheme. Overall, our work reduces costs and resource consumption and is desirable for preserving the battery energy of mobile devices.

Complexity Analysis

We measure the complexity of our proposed scheme in terms of computing time cost. The time requirement of our proposed scheme is briefly listed in Table 6, where the following notations are used:

- T_h: Time for performing a one-way hash function
- T_{code}: Time for generating a code for each photo
- $T_{Procedure}$: Time for performing any supported function to our work such as date, memory size, sort and filtering
- T_{Delete}: Time for performing deletion function of repeated photos in the MS
- T_{Oper}: Time for performing mathematical operations, such as summation
- T_{SIFC}: Time for performing creation function of SIF
- T_{SIFU}: Time for performing maintaining function of SIF
- N: Number of photos

Table 7 shows the time complexity for each algorithm.

Experimental Results

We evaluate the performance of our proposed scheme in terms of search efficiency, SIF maintenance, search precision, flexibility, effectiveness, battery power consumption and CPU performance. We report the practical results of our work on a real-world data based on coloured photos (10,000) saved in different folders, such as Viber, Facebook and WhatsApp, on a 'Samsung SM-G900H 4d0079e94bb9413f' smartphone. Furthermore, back up photos are uploaded to the CS.

Our experiments are conducted on a 2.2 GHz Intel i7- 4702MQ processor with a Windows 10 Home Single operating system of 64 bits (10.0, Build 16299) and 8 GB. Furthermore, the mobile device used is Samsung Galaxy S5(SM-G900H) with the following specifications: Board universal 5422, screen size of 5.01 inches, RAM of 16 MB android version 6.0.1, Boot loader G900HXXS1CQD1, Kernel Architecture ammv71 and Kernel version 3 10.9-7760371(G900HXXS1CQD1). We use Android Studio 3.0.1, Build #AI-171.4443003 (built on November 9, 2017), JRE: 1.8.0_152-release-915-b01 amd64 and JVM: OpenJDK 64-Bit Server VM by JetBrains to implement our experiments. Notably, the initial running time for creating SIF is 3,096,836,917 ns for 10,000 photos saved in the smartphone with the same features mentioned above, whereas the processing time for searching photos inside the SIF is 303,514,498.2 ns. In case of 372 photos of different sizes, the total average sizes of each photo and its unique code are 3,133.481 KB and 9 Bytes, respectively. Therefore, the main benefits of obtaining the photo's code are the high speed of retrieving the target photo and the high speed of applying the addition and deletion operations on the MM or SIF.

Table 4: Mechanism of addition operation on search index file and mobile memory

				Size of SIF and MM before applying the proposed scheme		Size of SIF and MM after applying the proposed scheme	
S	Day	No. of new photos	Processing time of adding photos to the device	SIF (KB)	MM (MB)	SIF (KB)	MM (MB)
1	18-2-2018	38	3863612583	316	1758	324	1900
2	19-2-2018	83	8417695204	324	1900	344	2203
3	20-2-2018	152	20778068706	344	2203	384	2751
4	21-2-2018	19	1943663917	384	2715	388	2792
5	22-2-2018	29	3447624087	388	2792	392	2905
6	23-2-2018	19	2106166504	392	2905	396	2982
7	24-2-2018	12	414064835	396	2982	498	3035
8	26-2-2018	31	3370840247	398	3035	404	3125
9	27-2-2018	41	512727585	404	3152	412	3306
10	28-2-2018	33	3914201127	412	3306	420	3430

Table 5: Mechanism of deletion operation on search index file and mobile memory

				Size of SIF and MM before applying the proposed scheme		Size of SIF and MM after applying the proposed scheme	
S	Day	No. of new Photos	Processing time of deleting photos from the device (ns)	SIF (KB)	MM (MB)	SIF (KB)	MM (MB)
1	24	2-2-2018	49712650	361	2135	355	2071
2	19	3-2-2018	39347931	355	2071	351	2019
3	12	4-2-2018	24791325	351	2019	348	1984
4	25	5-2-2018	51780593	348	1984	342	1915
5	10	6-2-2018	20709437	342	1915	339	1883
6	22	7-2-2018	45560802	339	1883	334	1822
7	29	8-2-2018	60057368	334	1822	327	1740
8	35	9-2-2018	72483031	327	1740	319	1646
9	9	10-2-2018	18638093	319	1646	317	1613
10	14	11-2-2018	28993212	317	1613	314	1571

Table 6: Computational cost

Phase	Computing
Initialisation Phase	$4\ T_{Oper} + 2\ T_{Procedure}$
Building Effective Search Index File Phase	$T_{SIFC} + T_{Procedure} + N\ T_{Code} + n\ T_h$
Provision of Storage Phase	$m\ T_{Delete} + T_{Procedure} + N\ T_{Code} + T_{SIFU}$
Sleeping Time Phase	$2\ T_{Procedure} + m\ T_{code} + m\ T_{Delete} + T_{SIFU}$
Total	$4\ T_{Oper} + 6\ T_{Procedure} + T_{SIFC} + 2\ T_{SIFU} + (2N+m)\ T_{code} + 2m\ T_{Delete}$

Table 7: Time Complexity of our proposed scheme for each algorithm

Phase	Time complexity
Algorithm 1	O(N)
Algorithm 2	$O(N^3)$
Algorithm 3	4 O(N) if the mobile has access to the Internet; otherwise, 2 O(N).
Algorithm 4	O(N)+ O(N log(N))
Algorithm 5	$O(N^3) + O(N)$

Table 8: Main standards of the proposed scheme

	Measurements	Before	After
Trial 1	Power Consumption	From 100% to 67%	From 100% to 73%
	Response Time (CPU)	1978600 ns	1768009 ns
	Storage Space	10146 GB	9305 GB
Trial 2	Power Consumption	From 100% to 70%	From 100% to 76%
	Response Time (CPU)	1984000 ns	1718417 ns
	Storage Space	10153 GB	9190 GB
Trial 3	Power Consumption	From 100% to 71%	From 100% to 76%
	Response Time (CPU)	1978800 ns	1608269 ns
	Storage Space	10019 GB	9150 GB
Trial 4	Power Consumption	From 100% to 65%	From 100% to 70%
	Response Time (CPU)	1899001 ns	1678296 ns
	Storage Space	9988 GB	8513 GB
Trial 5	Power Consumption	From 100% to 67%	From 100% to 75%
	Response Time (CPU)	1998139 ns	1768259 ns
	Storage Space	10344 GB	9492 GB
Mean	Power Consumption	From 100% to 68%	From 100% to 74%
	Response Time (CPU)	1967708 ns	1708250 ns
	Storage Space	10130 GB	9130 GB

Scenario

To achieve the objectives of the proposed work, we conduct a series of practical experiments on a 'Samsung SM-G900H 4d0079e94bb9413f' smartphone. The work scenario can be summarised as follows.

The smartphone contains a set of repeated photos in the MM or CS. All these photos have reached the phone through different social network applications, which may lead to repeated storage of the same photos in more than one place.

We rely on a set of measurements before and after the proposed work is applied. These measurements are the power consumption of the mobile phone, the response time to run an application from the moment it is touched (launch) to the moment of completion of the process (resume) and the calculation of the storage size. We implement the experiment five times then find the mean of results which considered the final result of our proposed scheme as shown in Table 8. The details of the experiments are before applying the proposed work, we conduct the following tasks. Power Consumption: We charge the device to 100% to determine the power consumption of the mobile device with repeated images and then leave the device for 24 h without charging. Response Time (CPU): We operate any application (from its launch to resume moment). Storage Space: We compute the total storage space. After applying the proposed work, we perform the following tasks. All duplicated photos have been removed. Power Consumption: We recharge the device to 100% and leave it for 24 h to detect the power consumption. Response Time (CPU) Storage Space Table 8 illustrates the potential and importance of the proposed work and the objectives achieved.

Conclusion

The mobile industry focuses on maintaining the efficiency of mobile devices in terms of storage and performance. Repeated photos received by these devices through social networks, such as Twitter and Facebook, consume much space of its memory and CS. In the long run, the processing and storage capabilities of these devices will slow down. Thus, we utilise a hash function-based SIF to design a scheme for removing duplicated photos from the device without affecting its capability to run the proposed scheme. The rest time of the device is intelligently determined using Algorithm 4. Our proposed framework for image recognition is based on building a unique code for each photo by generating a SIF and then deleting the repeated photos. The TT plays a main role in the performance of our work because it keeps the name, path and type of operation for removing all photos from the MM or CS. We use hash, ASCII and Weight functions to obtain a representative code for each photo. This code accumulates or retrieves duplicated photos in high performance and consequently decreases matching costs. Our formula implementation over 1,000 photos confirm that our proposed scheme results in only minimal losses in search time and precision compared with existing image search methodologies for plain texts. The proposed work also supports green computing by saving storage space through eliminating repeated photos and thus automatically increasing the CPU performance of the device.

Acknowledgement

The proposed study was conducted by computer science team in south of Iraq. This work was designed and implemented between the December 2017 and October 2018. The authors gratefully acknowledge the support of University of Basrah.

Author's Contributions

Ali A. Yassin: Coordinated the study, result analysis and full paper writing.

Abdulla J. Yassin: Software Programming and result collection.

Abdullah Mohammed Rashid: The legal research principles and supervised the draft manuscript.

Ahmed A. Alkadhmawee: Draft designing and contributed in data interpretation and figures designing.

Ethics

This article is original and contains unpublished material. The authors confirm that they have read the manuscript carefully and approve for publication. There are no ethical issues involved.

References

Akherfi, K., M. Gerndt and H. Harroud, 2016. Mobile cloud computing for computation offloading: Issues and challenges. Applied Comput. Informatics, 14: 1-16. DOI: 10.1016/j.aci.2016.11.002

Beloglazov, A., J. Abawajy and R. Buyya, 2012. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Generat. Comput. Sys., 28: 755-768.

Danova, T., 2015. The smartphone report by country: Adoption, platform and vendor trends in major mobile markets around world.

Gai, K., M. Qiu, H. Zhao, L. Tao and Z. Zong, 2016. Dynamic energy-aware cloudlet-based mobile cloud computing model for green computing. J. Network Comput. Applicat., 59: 46-54. DOI: 10.1016/j.jnca.2015.05.016

Hurbungs, V., Y. Beeharry, A.K. Calkee and G. Ahotar, 2016. An energy efficient android application. ADBU J. Eng. Technol.

Korhonen, K., 2011. Predicting mobile device battery life. PhD. Thesis, Department of communication and networking, Espoo.

Lewis, G.A., S. Echeverría, S. Simanta, B. Bradshaw and J. Root, 2014. Cloudlet-based cyber-foraging for mobile systems in resource-constrained edge environments. Proceedings of the 36th International Conference on Software Engineering, May 31, Hyderabad, India, ACM, pp: 412-415. DOI: 10.1145/2591062.2591119

Lu, X., H. Hassanein and S. Akl, 2005. Energy aware dynamic task allocation in mobile ad hoc networks. Proceedings of the International Conference on Wireless Networks, Communications and Mobile Computing, Jun. 13-16, IEEE Xplore Press, Maui, HI, USA. DOI: 10.1109/WIRLES.2005.1549465

Lundquist, A.R., E.J. Lefebvre and S.J. Garramone, 2014. Smartphones: Fulfilling the need for immediacy in everyday life, but at what cost. Int. J. Humanities Social Sci., 4: 80-89.

Magurawalage, C.M.S., K. Yang, L. Hu and J. Zhangc, 2014. Energy-efficient and network-aware offloading algorithm for mobile cloud computing. Comput. Networks, 74: 22-33. DOI: 10.1016/j.comnet.2014.06.020

McDaniel, B.T. and S.M. Coyne, 2016. Technoference: The interference of technology in couple relationships and implications for women's personal and relational well-being. Psychol. Popular Media Culture, 5: 85-98. DOI: 10.1037/ppm0000065

Misra, S., L. Cheng, J. Genevie and M. Yuan, 2016. The iPhone effect: The quality of in-person social interactions in the presence of mobile devices. Environm. Behavior, 48: 275-298. DOI: 10.1177/0013916514539755

Mok, J.Y., S. Choi, D. Kim, J. Choi and E. Choi, *et al.*, 2014. Latent class analysis on internet and smartphone addiction in college students. Neuropsychiatric Disease Treatment, 10: 817-828. DOI: 10.2147/NDT.S59293

Roy, A., S.M. Rumble, R. Stutsman, P. Levis, D. Mazières and N. Zeldovich, 2011. Energy management in mobile devices with the cinder operating system. Proceedings of the sixth conference on Computer systems, Apr. 10-13, ACM, Salzburg, Austria, pp: 139-152. DOI: 10.1145/1966445.1966459

Saha, B. and D. Srivastava, 2014. Data quality: The other face of big data. Proceedings of the IEEE 30th International Conference on Data Engineering, Mar. 31, IEEE Xplore Press, Chicago, IL, USA. DOI: 10.1109/ICDE.2014.6816764

Schober, M.F., J. Pasek, L. Guggenheim, C.1. Lampe and F.G. Conrad, 2016. Social media analyses for social measurement. Public opinion quarterly, 80: 180-211. DOI: 10.1093/poq/nfv048

Shah, D.V., J.N. Cappella and W.R. Neuman, 2015. Big data, digital media and computational social science: Possibilities and perils. ANNALS Am. Acad. Political Social Sci., 659: 6-13.

Shen, Y., H.C. Chan and C.S. Heng, 2016. The medium matters: Effects on what consumers talk about regarding movie trailers. Proceedings of the 37th International Conference on Information Systems, (CIS' 16), Dublin, pp: 1-11.

Shiraz, M., M. Sookhak, A. Gani and S.A.A. Shah, 2015. A study on the critical analysis of computational offloading frameworks for mobile cloud computing. J. Network Comput. Applicat., 47: 47-60.

Shiraz, M., M. Whaiduzzaman and A. Gani, 2013. A study on anatomy of smartphone. Comput. Communicat. Collaborat., 1: 24-31. DOI: 10.1016/j.comnet.2014.06.020

Sprecher, S., 2016. Can i connect with both you and my social network? Access to network-salient communication technology and get-acquainted interactions. Comput. Human Behavior, 62: 423-432.

Verbelen, T., P. Simoens, F. Turck and B. Dhoedt, 2012. Cloudlets: Bringing the cloud to the mobile user. in Proceedings of the 3rd ACM workshop on Mobile cloud computing and services. Jun. 25-25, Low Wood Bay, Lake District, ACM, pp: 29-36. DOI: 10.1145/2307849.2307858