An Optimal Data Access Framework for Telerehabilitation System

Abdullah Muhammed[2], Waidah Ismail[1,3*], Siti Nabilah Basarang[1], Ali Y. Aldailamy[2], Rimuljo Hendradi[3]
[1] Faculty of Science and Technology, Universiti Sains Islam Malaysia, Nilai, Negeri Sembilan, Malaysia.; waidah@usim.edu.my.
[2] Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, (UPM), Serdang, Selangor, Malaysia.; a.aldailamy7@gmail.com; abdullah@upm.edu.my
[3]Information System Study Program, Faculty Science and Technology, Universitas Airlangga, Kampus C, Surabaya, Indonesia; rimuljohendradi@fst.unair.ac.id
Corresponding author: waidah@usim.edu.my

Abstract—In the telerehabilitation system, the statistical data of the patients' movement are stored in the temporary storage and synchronised to the storage service of online cloud data. Application providers faced a problem in reducing the monetary cost of the whole cloud service and reducing the footprint of the main memory space. In addition, users encounter long latency when the required data need to be read from the cloud via the internet and the hard disk drive (HDD) of the cloud servers. To solve this problem, an optimal data access framework is presented to cache the statistical data of the patients in the application server. The main memory database and cache use internal tracking in the main memory to track records that are not accessed by transferring the data to the disk. This mechanism retains the keys and all indexed fields of evicted records in the main memory which prevents potential memory space savings for the application that have many keys and secondary indexes. Therefore, to overcome the mentioned problems, the cloud database is categorised into three partitions (hot, warm, cold). In addition, a cache memory image in the application server is provided for the hot partition of the cloud database. The use of cache memory image reduces the number of reading operations from the cloud and saves the space of the main memory. The experimental results showed that the proposed framework can produce good quality solutions by utilising the main memory space and reducing the latency and read operations from the cloud that lead to reducing the monetary costs.

Index Terms— Rehabilitation system, exergame, cloud computing, DBaaS, cache memory.

I. Introduction

There has been a drastic increase of data in the science field, particularly in the health care field. It is believed that the massive data in health care is the result of the advancements in health care technologies [1, 2]. Tele-rehabilitation is one of the important branches of health care in helping patients to restore the highest level of quality, independence, and function of life after certain disabilities [3, 4]. Tele-rehabilitation is defined as an alternative against the usual rehabilitation service which helps patients in rural areas to get access to less service that is practical concerning logistics and costs in a controlled environment. It is including the usage of mobile phones or other wireless devices as applied to rehabilitation exercises. Applications or software of such include exercises in games, treatment monitoring based on their rehabilitation progress and data analysis [5-7].

Valentina et al. [8] introduced virtual rehabilitation (VR) that stores, processes, and analyses massive data to monitor the improvement of patients. The advancements in data management systems such as cloud computing and virtualisation allow the development of systems that provide effective storage and manipulate a large amount of data [9-11].

In the cloud-based systems, the results of the patients' exergames are uploaded on cloud and the therapist can access the patients' information and assess their movement improvements. There are two-fold obstacles when transferring data from the cloud to the application server namely latency and read cost from cloud [13-15]. The latency problem is due to reading the full history of patients' statistical data from the cloud. Furthermore, cumulative analysis has to read massive statistical data from the cloud which is time-consuming for the end-users. When the statistical data of a patient is required for analysis, a read operation for all patient statistical data is usually sent to the cloud that can lead to the data retrieving cost problem [16]. According to Mansouri et al [17] and Zhang et al. [18], the monetary cost is charged for the get operation and network consumption when reading patient statistical data from the cloud. Therefore, the more the data is read from the cloud, the more monetary cost is charged.

The aforementioned problems can be solved by caching the statistical data of the patients in the application server. There are two levels of caching in the application server, namely hard disk drive (HDD) and memory caching [19, 20]. Regardless of the type of HDD (e.g. SATA or SSD), reading and writing operations on HDD are very slow in comparison with memory [21]. Thus, the cache memory copy of the stored data in the HDD can enhance the performance [22]. The cache memory copy allows the analysis application to read data directly from the local memory that reduces the latency when reading the data via the internet from the cloud hard disks. Reading the data from the memory is 1,000 times faster than reading from HDD [20]. Therefore, keeping the data in the memory can minimise the latency by reading the data that are stored in the hard disk. Besides that, reducing the number of reading operations from the cloud can minimise the overall cost of the cloud service.

Moreover, in the memory cache, the available main memory is insufficient to store a large database, which is the main problem of the technique. The problem can be overcome using a memory eviction algorithm to manage the main memory space allocated to the cache [23]. The eviction algorithm transfers some of data to other space for new records and selects which data should be kept and release to the disk in reducing the memory footprint. Examples of the eviction algorithms are least-recently used (LRU) [24] and least frequently used (LFU) [25]. However, some issues make the eviction algorithms less efficient in terms of performance and memory size management [26-28]. Firstly, the cache eviction algorithm uses an internal tracking mechanism to identify records, which are not accessed for eviction. The

cache keeps the corresponding internal tracking for subsequent tracking even after removing the records that were not accessed from memory. This internal tracking takes up memory space and charges extra cost on memory management. Second, all available database caching and cache memory database systems retain the keys and indexed fields of the evicted records. This step reduces potential memory capacity for platforms with many secondary indexed fields.

In order to tackle these issues, the transferred data in the main memory should be filtered in the database of the disk. Therefore, in this study, the database is divided into three partitions according to the status of the patient as follows: 1) hot/active data that belong to active patients; 2) warm/inactive data that belong to the patients who are not currently active or have been discharged; and 3) cold/dead data that belong to the patient with inactive status; for example, dead patients. Only the data that belong to hot partition were moved to the main memory. The data of the patients are moved to a different partition in the database according to the new status and the change is synchronised to the main memory copy.

In summation, in order to help fast retrieval of the medical data in the huge dataset, an optimal data access framework is proposed in this paper. In the presented scheme, in order to solve the delay and reading cost problems, caching technique is performed on the statistical data. In HDD caching technique, in order to overcome the latency problem in reading and writing operations on HDD, the proposed technique uses a cache memory copy of the stored data in the HDD, which leads to reduce latency and the data reading monetary cost. On the other hand, to tackle the insufficient main memory problem in the memory caching and to manage the main memory space allocated to the cache, the eviction algorithm is utilized. Internal tracking is considered as a drawback in eviction algorithms, which can be overcome by partitioning the database based on the patients' status. The main contributions of the proposed scheme are as follows:

i. Unlike the previous cloud-based scheme, in this paper, the statistical data of the patients are cached in the application server, which leads to overcome the delay problem and reduce the monetary costs when downloading data to the application server from the cloud.

ii. Unlike the previous works, the cache memory image of the hot partition in the application server-side is utilized which leads to improve database access time and lessen the number of reading operations issued to the cloud.

The subsequent sections of this paper are as follows. Section 2 reviews the related works. Section 3 presents the methodology. The mathematical model is presented in section 4. Section 5 presents the dataset, experiment setup, and benchmarking results. Section 6 provides the discussions on the results. Section 7 summarises the findings.

II. RELATED WORK

In the last decades, several schemes have been proposed to Goli-Malekabadi et al. [12] suggested a cumulative analysis of data to define the best procedure for a group of patients that can find effective treatments, identify unknown relationships, recognise new patterns, and make universal decisions. Cumulative analysis requires a great amount of data movements from the cloud to the application server (analysis server), which processes the data and produces cumulative results.

In [51], authors claimed that load balancing is a technique of distributing the loads among various virtual machines to minimize the response time, minimize the cost, minimize the resource utilization, and minimize the overhead. They attempted to experimentally verify how to minimize the response time and processing time through the tool named cloud analyst.

In [52], a cloud robotic system was proposed to provide an Internet-based process for upper-limb rehabilitation with multimodal interaction. Is such system, the therapist can remotely supervise the patient to exercise and set the rehabilitation training mode, therapy game level, and control the parameters of rehabilitation robot. The physiotherapist can also on-line evaluate the current performance of the patients according to their history of data, the dialogue results and the advising from the central server over networks. Based on the evaluation results, the proposed system is cost-efficient and more convenient for both the patients and the therapist.

In addition, a rehabilitation training and administration system (TAS) for upper limb rehabilitation robot is presented in [53]. Patient information interfaces are introduced so that physiotherapists can collect, store, retrieve and exchange patients' information electronically. likewise, the authors have programmed a passive training method and virtual reality exergame in order to stimulate users' enthusiasm in training process. Based on the evaluation results, TAS could improve the efficiency of physiotherapies with providing a feasible solution to the shortage of rehabilitation therapists.

Moreover, the authors in [54] developed a novel context-based adaptive home-based rehabilitation (CAHR) framework which aimed to optimize recovery based on patient's performance and ambient conditions. In addition, the authors modeled the adaptation mathematically in order to configure the rehabilitation task based on patients' conditions. Results indicated that the CAHR can beneficial by improving the patients' performance. Furthermore, a tele-rehabilitation system was presented in [55] to enhance community rehabilitation for patients who are discharged early from the hospital. Likewise, in [56], a self-adaptive and hybrid mode with both patient self-management and doctor remote diagnosis based telemedicine framework was presented. The scheme included two parts: intelligent diagnosis and real-time diagnosis. Generally, a patient could use intelligent diagnosis to take a self-service diagnosis first, if the patient satisfies output, real-time diagnosis is passed. Otherwise, the patient could use real-time diagnosis to take a manual diagnosis. The combination of intelligent way and real-time way reduced doctor's work amount in diagnosis.

Based on the aforementioned discussion, none of the previous cloud-based schemes have utilized the cache memory, which leads to occur the delay problem and increases the monetary costs when transferring data from the cloud to the application server. Therefore, presenting a feasible method to overcome the aforementioned problem is vital. Accordingly, in this paper, an optimal data access framework is presented to cache the statistical data of the patients in the application server. Moreover, in order to manage the main memory space allocated to the cache, the memory eviction algorithm is utilized. However, in the eviction algorithm, internal tracking is used which leads to occupying the memory space and charges extra cost on memory management. To overcome the mentioned drawback, the data portioning approach is utilized.

III. METHODOLOGY

In this section, the proposed optimal data access framework is presented. The schematic diagram of the proposed method is shown in Figure 1.

A. Architecture of the proposed framework

The main utilized component of the proposed method consists of medical interactive rehabilitation assistant (MIRA), MySQL Database, and Apache Ignite. These components are described in detail in the next sub-sections.

a. MIRA

MIRA is a healthcare solution that conducts the rehabilitation process using a virtual reality technology [29, 30]. It is a game-based solution designed to provide meaningful feedback such as statistical data about the patients' movements from the Kinect device that is equipped with a motion-tracking sensor [31-33]. The sensor aids the system in capturing important attributes (data) of the patients' movements such as average percentage, average speed, average acceleration, distance, and points for the evaluation of their performance [34]. However, MIRA system currently lacks the tools in providing a meaningful evaluation of patients' performance to the physiotherapist [32]. MIRA produces a big amount of statistical data for every session. It stores all the statistical data of patients' movements into the Microsoft Azure cloud [35]. An analysis application is needed to read the statistical data frequently from the cloud for further analysis to provide meaningful data. Therefore, this study proposed a framework that provides an effective and efficient patient data analysis module for better evaluation of patients' performance such as performance prediction [12], preference optimisation [13], and preference recommendation [14].

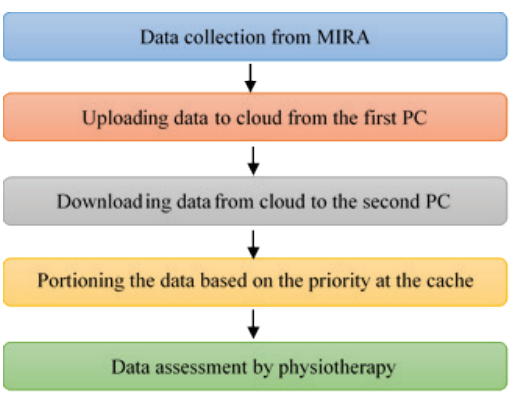


Fig.1. The schematic diagram of the proposed method

b. MySQL Database

MySQL DBMS is used to create databases in the separate server called database server [36]. The database server has a separate database for MIRA users to store the data retrieved from MIRA cloud. The databases store images of the MIRA cloud retrieved data that have many differences in their structures as follows:

- Data retrieved from the MIRA cloud are normalised and spread among multiple tables to eliminate redundancy of the data.
- Fixed textual values such as gender, difficulty, and side retrieved from the MIRA cloud are coded into numeric values to reduce storage space occupied by text values.
- There are three tables for every entity in every database. The tables present the status of the patients and they are captioned by the statuses, namely hot, warm, and cold tables which contain the data of active, inactive, and dead patients, respectively. Therefore, the data of the patients are moved in accordance with the patients' status. The movement of patients' data is further discussed in the next sections.

c. MySQL Database

Apache Ignite is a cache memory computing framework that develops a highly available and strongly consistent cache memory database or cache [37, 38]. It is a high-performance computing framework used for the cache memory processing of large-scale and real-time data sets. It looks like a distributed data storage that works as a cache memory database [37]. This study used Apache Ignite to create the hot partition image in the main memory of the application server after reading the data from the MIRA cloud. The cache memory image of data is used to cache the data of currently active patients to provide better reading performance. The fast read allows real-time analysis on preference recommender, preference optimisations, and decision tree analysis.

B. Proposed Optimal Data Access Framework for Telerehabilitation System

In this paper, the used data are the results of MIRA games. The statistical data of the patients' movement are stored in MIRA application temporary storage before the MIRA application synchronises the data to the database-as-a-service (DBaaS) in the MIRA cloud. The application server has two types of data download, namely manual and automatic [39]. The manual download is provided to the users to download the data immediately for usage. The automatic download is a time-based download that can be set by the users. For both methods, the application server connects to the MIRA cloud using a username and password which return an authorised access token. Then, the application server uses the token with startDate and endDate to download the statistical data of the patients' movement. The startDate and endDate are used to download the data that are added to the DBaaS of the cloud from the last download to the current time of the application server. This mechanism ensures the non-duplication of the downloaded data. Then, the data are downloaded in CSV file format which is compressed in a ZIP file format. The application server expands the zipped file and extracts the CSV file. Finally, the process continues with the cleaning operation of the raw data in the CSV file and produces data that is suitable for both the main memory and disk-based database.

In the proposed framework, the MIRA application writes the data to cloud DBaaS that stores statistical data on patients' movements. The analysis applications read the data from the MIRA cloud to analyse, evaluate, and produce the results of patients' progress. The analysis applications run many read operations to data stored in MIRA cloud, which increases the monetary cost. In order to minimise this cost, this study

proposed an application side cache memory image of database for the current active patients. The application side cache memory image of the database is developed using Apache Ignite. In the next sub-section, the proposed framework and contributed functions are described in detail.

Figure 2 shows the flow of the data in the proposed framework in details. The architecture of figure 2 shows the patients who are playing the MIRA (Exergames) in the Rehabilitation center, where all the movement will be capture in the PC or notebook. Every10 minutes the synchronization happens in the MIRA cloud where all the data will be kept. The MIRA data is downloaded every day around 12 PM to ensure the medical doctor or physiotherapy can view the data automatically the next day. Downloading the data is performed manually or automatically and it will be downloaded using a zip file and then will be unzipped under CSV. In addition, the data can be cleaned and populated into a database server, which is divided into three parts called Hot partition, Warm partition, and Cold partition.

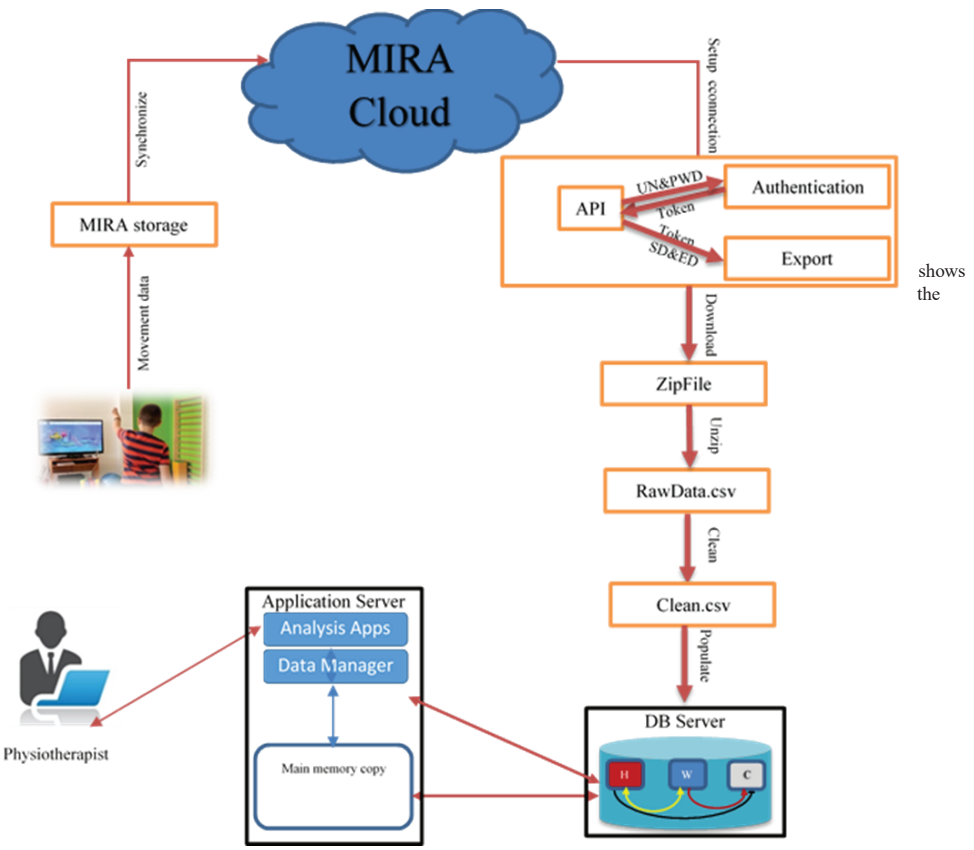
The data of the active patients are stored in the hot partition. The status of a patient can be changed to warm if the respective data has not been accessed for 3 months or the patient has been discharged from the hospital. The data of the inactive patients are stored in the cold partition.

In cache memory data, the application server needs to have a data analyst and data manager in analyzing data and organizing the data in retrieving the cache memory. Finally, the physiotherapies can access the data.

a. Data Normalisation and Coding

The data are stored in the CSV file as they are retrieved from the cloud. The CSV file is structured as one table that is not normalised and has many fixed text values which can be coded into numeric values. Normalisation. All the data are downloaded in one table that has several redundancies in each record such as PatientID, FirstName, LastName, BirthDate, and others. Therefore, normalisation is used as a systematic approach to produce multiple tables to eliminate data redundancy (repetition). Figure 2 shows the six levels of normal form, and the best level is in the 3rd normal form for most practical applications.

After downloading and extracting the data, each column in the download file has a table column in the database. The data in this column are inserted into the corresponding table column and the redundant data in the column are eliminated. The database consists of a set of relational tables that eliminate any redundancy. Coding. For the optimisation of the storage space, the textual data that have fixed values such as gender, side, difficulty (e.g., easy, medium, and hard), and diagnosis are converted into numbers to reduce storage spaces and computation overhead of string values. For example, Table 1 coding for the values of the difficulty column in the database.



shows the

Fig.2. Data flow of the proposed framework

Fig.3. Normalisation forms

b) Database Custom Partitioning

The database is divided into three partitions according to the patients' status. The three statuses include active

Table 1. Example of data coding

Textual	Code
Easy	1
medium	2
hard	3

patients, inactive patients, dead patients whose data are placed in hot partition, warm partition, and cold partition, respectively, as shown in Figure 2. The data of active patients are considered as hot data and they are stored in the partition called hot. The hot partition has an identical copy that is always stored in the main memory. Any active patient data can be moved to warm partition for cases where the patient is discharged or stops the therapy exercises. The data can be moved to cold partition for dead patients. When the data are moved to the warm partition or cold partition, they are completely deleted from the hot partition and also from their corresponding main memory image. The inactive status of the patient can be changed to active or discharged at any time. After changing the status of the inactive patient, the data are immediately moved to the corresponding partition of the new status. The cold partition contains the data of the dead patients and their status cannot be changed to other statuses. Any changes in the hot partition such as adding or deleting patients' data are immediately synchronised with the copy in the main memory. All of the operations of data movement and the manipulation of hot partition are performed by the data manager in the application server-side.

The custom partitioning filters the data in the disk-based database. Therefore, only the hot data have an identical copy in the main memory. In this case, the memory for the evicted data of inactive and dead patients removes the internal tracks, keys, and indexed fields. The internal tracks, keys, and indexed fields are deleted from the hot partition copy in the main memory when the patients' data are moved from the hot partition. When the patients' data are moved (for example, inactive patient becomes active) or added (registration of a new patient or current active patients play more games) to the hot partition, all the information are synchronised to the hot partition image in the main memory of the application server-side.

b. Main Memory Copy of Hot Partition (Cache)

The proposed solution provides a database caching system that can cache a huge amount of data like the cache memory database while allowing the data manager to choose which data that need to be cached (not all the database such as the cache memory database). The main memory copy is based on ACID (atomicity, consistency, isolation, durability) which provides strong consistency of the data. Moreover, it keeps all the calculated and derived values. Hence, there is no need to recalculate and derive the data whenever they are required by users.

The main advantage of the proposed main memory copy of the hot partition is its selectiveness. The proposed solution allows caching the currently used data (data of active patients) to the main memory, unlike the cache memory databases that store the entire copy of the database in the main memory. Caching is only done on the data that need to be cached to increase memory size savings. Another advantage is the reduced database lookups overhead as the data manager knows which partition contains the required data. This step is performed by adding one column to the patients' data for the current status that tells the system the partition of the patients' data. Besides that, the hot partition is larger than the available cache space. LRU algorithm is used to manage the data in the cache and decides which record should stay in the cache and which one should be evicted to the hot partition in the disk-based database. It also minimises the number of reading operations from the MIRA cloud that reduces the monetary cost of the whole service. It is believed that the proposed solutions can solve the above-mentioned problems.

c. Cloud Service Cost

When the patients' data are required, a read operation is sent to the MIRA cloud. Every time the data are read from the cloud, the monetary cost is charged for reading operations and network consumption. Therefore, the more the data are read from the cloud, charges more money. The proposed system reads any piece of data only once before storing the data in the application server-side in the main memory copy of hot partition for any subsequent access. Whenever there is a read request of data, the proposed system searches for the requested data in the main memory copy of the hot partition. If the data are not in memory, the data can be retrieved from the cloud. In order to provide optimised read operation, the system performs event-derived synchronisation of the hot partition in the disk-based database and the main memory copy in the application server-side. It allows users to set periodic and manual download from the cloud. This mechanism reduces the number of reading operations from the cloud, which charges the monetary cost for every read operation, network consumption, and data retrieval.

IV. MATHEMATICAL MODEL

This section discusses all the achieved optimisations of the proposed framework, and the modelling part is conducted to derive a mathematical equation that shows how latency, cost, and memory savings are optimised. Developed mathematical equations are the objective function for the process of optimisation. The mathematical models for every optimisation are as follows:

A. Access Latency Optimisation

The data access latency for a client is caused by cloud (network), hard disk drive, and main memory latencies, which are discussed in detail in the next three subsections.

Cloud latency: In the cloud, access latency is determined according to the suspension of time when the request is issued and the time when that data is obtained. Small size data latency is governed by network latency, and the latency is estimated

using the round trip time (RTT) [40] between the cloud (the source) and application server (destination) and vice versa that is calculated according to Eq. (1).

$$RRT(ms) = 0.02 \times \text{distances}(km) + 5 \quad (1)$$

Where, 0.02 is the latency for each kilometer in which distance(km) denotes the distance between the client and the server, and the estimated time for routing latency is 5.

HDD latency: HDD latency is measured using spindle speed, which is usually given in rotations per minute [41]. HDD latency is usually measured in the millisecond and calculated using Eq. (2):

$$ALHDD = \left(1/\left(SpindleSpeed/60\right)\right) \times 0.5 \times 1000 \quad (2)$$

Where ALHDD is the average latency of the hard disk drive. Moreover, the numbers of tracks to read are 1, 60, 0.5, and 1,000 for the seconds of a minute, time to move the head, and time needed to navigate the spindle to another track. SpindleSpeed denotes the number of spindle rotations per minute.

Memory latency: Memory access latency [42] is always measured in the microsecond and calculated using Eq. (3).

$$ALMEM = h \times Tc + (1-h) \times M \quad (3)$$

Where ALMEM shows the average latency of memory. Furthermore, h, (1-h), Tc, and M represent the hit rate, the miss rate, the time to access information from the cache, and the miss penalty (time to access main memory). It is necessary to consider the three types of latencies and how many times each latency is met to measure the total time for a client to read the data from the cloud. Therefore, the total latency for the request issued to the cloud through the application server is calculated using Eq. (4).

$$TLatency = 4 \times RTT + 2 \times ALHDD + 3 \times ALMEM \quad (4)$$

Where TLatency is the total latency of client request. Likewise, the optimised total latency for a client in reading the data from the cache of the application server-side is measured using Eq. (5).

$$OTLatency = 2 \times RTT + 2 \times ALMEM \quad (5)$$

B. Cost Optimisation

The full history of the patients' statistical data is read from the cloud for every evaluation of the patients' progress by the physiotherapist. This step increases the cloud service cost and access latency. Therefore, MIRA analysis of the data retrieval system downloads the data only once. MIRA analysis server has two types of data downloading, which are manual and automatic. In the manual download, users can download the required data immediately, whereas automatic download is a time-based download that can be set by the users and run automatically on time. In automatic download, every single record of the data is downloaded only once before being stored in the database server for any subsequent access by the physiotherapist. The proposed mechanism guarantees no repeated reading of the record from MIRA cloud DBaaS.

The data manager searches for the requested data in the cache when the physiotherapist requested the data. If the data are not in the cache, the data can be retrieved from the database of the local database server. If the data do not exist in the cache and database, users have to click on the download button to retrieve data from the MIRA cloud. After retrieving the data from the cloud, they are stored in the database and cache for any subsequent access.

A read operation for all historical data of the patient is sent to the cloud when the patients' data is required for analysis. Every time the data is read from the cloud, the monetary cost is charged for reading operation and network consumption. Thus, more money is charged when more data are read from the cloud. The monetary service cost is calculated using two separate costs, which include read cost and network consumption cost, and it will be discussed in the next two subsections.

Read cost. In the Microsoft Azure cloud, every read operation is charged to the user. The cost is determined by multiplying the number of reading operations with the reading cost of the data center of the cloud as shown in Eq. (6):

$$R\cos t = Rno \times R\cos t \quad (6)$$

Where Rcost denotes the total cost of the total number of reading operations in a month. Rno is the sum of reading operations per month, and Rcost shows the cost applied by the cloud for every read operation.

Network cost. The transfer of data from the cloud to the client has to be paid by the user. For each gigabyte, there is a monetary cost. Therefore, the reduction of data transferred throughout the cloud network has a big impact on the overall cost of the cloud service. The cloud network consumption cost is calculated using Eq. (7).

$$N\cos t = Rno \times Dsize \times T\cos t \quad (7)$$

Where Ncost denotes the total cost of the total number of data transfer operations in a month. Rno shows the sum of reading operations per month. Dsize is the total data size transferred throughout the cloud network and Tcost denotes the cost applied by the cloud for every gigabyte of data transfer.

The overall cost of the cloud service is the summation of the read cost and the network consumption cost, which is calculated as the following:

$$OT\cos t = R\cos t + N\cos t \quad (8)$$

The costs above are charged for every read operation of the patients' statistical data. However, every data record is only read once from the cloud using the cache in the application server-side. Every record of the data is read for the first time before it is kept in the cache of the application server-side for any subsequent access by the client. The subsequent accesses are not charged for any monetary cost as they are performed from the cache. Therefore, caching optimises the cloud cost, and the optimisation is calculated using Eq. (9):

$$OT\cos t = R\cos t + Dsize \times N\cos t \quad (9)$$

Where OTcost is the total cost after optimisation. The size of the downloaded data is added to X, which is the accumulated value for the data size downloaded from the cloud, to measure the cost optimisation performed by the cache. The number inside X increases according to the data size of the current download that increases for every data downloaded from the MIRA cloud.

The number inside X increases when the patient plays more games until the status is changed to inactive or dead. Then, the value of X is stored in the database. Even when the patients' status is active, all of the old statistical data are stored in the database and they are automatically moved to the cache. Thus, the new downloads of the patients' data begin from the last point of the saved data.

The formula to measure the cost saved in every data downloaded from the cloud is as follows. We applied the transferred cost from Eq. (2) of the cloud for the data size stored inside X using Eq. (10) to calculate the total cost of the downloaded data without considering the cached data.

$$C = \sum_{k=0}^{n} X^k \times N \cos t \tag{10}$$

By considering that the old data are downloaded and kept in the cache, the system only downloads the newly added data. Therefore, Eq. (11) calculates the download cost for the newly added data that are not previously downloaded. The accumulative download of data from the cloud ensures that the system does not download the same data in every access of data. Every time the download operation takes place, it retrieves the data from the last point of time when the data were downloaded.

$$Coptimized = \sum_{k=0}^{n} \left(X^k - X^{k-1} \right) \times N \cos t \tag{11}$$

C. Memory Space Optimization

There are two-fold issues when employing eviction algorithms in managing the memory space: 1) the cache still preserves internal tracking of evicted records; and 2) it retains keys and indexed fields of records after eviction. This study used Apache Ignite library to create the cache memory image of the hot partition. However, Apache Ignite uses LRU algorithm as the main eviction technique for memory space management. LRU algorithm has the two-fold issues that waste the memory capacity [38].

It is necessary to avoid moving the data of cold and warm partitions to the cache in preventing the cache from creating internal tracks that keep the keys and indexed fields after they are evicted from the memory. When they become hot data, they are moved to the memory and the cache creates an internal track for them. After they return to cold or warm status, the cache deletes them from the memory including the internal tracking, keys, and indexed fields.

The optimisation of memory space management can be achieved through database partitioning that sorts the data to active (hot), inactive (worm), or dead (cold). The partitioning of data according to the patients' status is performed by the data manager, which is an intermediate layer between the analysis application from one side and cache with the database from the other side. It transfers the patients' data between the partitions according to their status. It also removes all the related internal tracks for all the patients' records after the status is changed from active to inactive or dead.

LRU algorithm creates an internal tracking to identify which records are not accessed anymore and select them for eviction to the disk-based database. Even after removing all records from the memory, the corresponding internal trackings are retained in the cache for future tracking. In this case, the number of records is different from the number of internal tracks, keys, and indexed fields. The occupied space of the allocated main memory to the cache is measured using Eq. (12).

$$\sum_{i=0}^{n} SizeOf(R_i) + \sum_{j=0}^{k} SizeOf(IT_j) + SizeOf\left(keys_j + indexes_j\right) \tag{12}$$

Where, Ri is the size of the record i, i∈ [0-n]. Likewise, ITj shows the size of the internal track j, j∈ [0-k]. Keysj is the size of the keys and indexed fields j, j∈ [0-k] and Indexesj denotes the size of the indexed fields j,j∈ [0-k].

This internal tracking increases the operating cost of memory management and lessens the evicted memory space. Moreover, LRU algorithm retains the keys and indexed fields of the evicted records that reduce the memory capacity for databases that have many indexed fields.

The proposed cache system reduces the overhead by moving the management of memory to the database level and the patient's data that are not accessed anymore to warm partition and cold partition when the patient is dead. The cache contains an exact copy of the data in the hot partition. Once the data are moved from hot partition, they are completely deleted from the main memory image, including the internal tracks, keys, and indexed fields. Any changes in the hot partition, such as adding or deleting patient data, are immediately synchronised with the copy in the main memory. Removing records with their internal tracks, keys, and indexed fields can increase the potential savings of the memory spaces. In this case, the number of records is the same as the number of internal tracks, keys, and indexed fields.

In order to measure the optimisation of memory space, we introduce Eq. (13) which calculates the size of all records and their internal tracks, keys, and indexed fields.

$$\sum_{i=0}^{n} SizeOf(R_i) + SizeOf(IT_i) + SizeOf\left(keys_i + indexes_i\right) \tag{13}$$

Generally, memory space optimisation is based on saving memory spaces that are wasted by eviction algorithm such as LRU and LFU algorithms [43]. For example, if the data is frequency accessed by the physiography, the location of the data will be the hot partition, which leads to reduce the memory space as the data will be in the cache memory of hot partition. This will save memory space optimization.

V. EXPERIMENTS, RESULTS AND DISCUSSION

This section discusses the dataset, experiment setup of the proposed framework, and results. The experiments aim to evaluate the proposed framework from the perspectives of latency, the monetary cost, and memory savings optimisations. The experimental study performs the comparison using cache memory and database. By performing the database it will take a longer time in accessing with involve three partitions (Hot, Warm and Cold), which will be discussed in detail in the next subsections.

A. Dataset and Experiment Setup

This study used real patient data from Perkeso Tun Razak Rehabilitation Centre for the implementation process. The patients played MIRA games from the year 2018 to 2019. The number of patients is more than 145 in which the minimum number of sessions for each patient is two and the maximum number of sessions is 547. The total number of records is 5,541 in which each record consists of 41 fields that have different numbers, text, and dates. The data are prepared to fit to the proposed approach table. Data preparation includes distributing the data of each session in one record by transforming the values of some columns to fields, coding some text values, and converting all date formats to one uniform format.

B. Latency Experiment

For the measurement of latency optimisation, this study conducted a real latency experiment using real-world MIRA cloud in Finland, MIRA analysis application and database server located in Nilai, Malaysia, and the client of Perkeso Tun Razak Rehabilitation Centre located in Melaka, Malaysia. The experiment is conducted using the proposed MIRA analysis application and patients' data from Perkeso. Perkeso Tun Razak Rehabilitation Centre has one MIRA terminal and receives an average of ten patients daily.

This study conducted a real latency experiment using real-world application server and database servers to measure latency optimisation. The same amount of data is read from the database that is stored in the HHD of the database server and the cache that is stored in the main memory of the application server. The experiment is conducted using external client and MIRA analysis application that use two tables from the database, the images in cache, and the details as shown in Table 2.

Table 2. Details of database tables

Table name	Preference	All_Var_Sessions
Columns number	6 columns (4 numbers, 2 strings)	44 columns (34 numbers, 3 dates, 7 strings)
Records number	2,000	2,000

Figure 4 shows the reading and writing operation of the data. The purpose of this measurement is to show the speed of reading and writing operation on the cloud after utilizing the caching system, which leads to improve the system performance. The comparison is based on two sections: preferences patients' records and all patients' records. Preferences data is those that need to perform algorithms for calculation. Preferences operation is performed only at the hot partition. From the comparison, it is shown that using cache memory performs better than the other one.

Figure 4.a shows that the cache has an advantage over the disk-based database for data writing. The data writing in the cache is faster by more than 5.5 times of writing operations compared to the disk-based database. For the All_Var_Sessions, the writing operations of the cache are faster than disk-based database by more than four times. The superiority of the cache in the writing operations is because the main memory (RAM) is faster than the hard disk (HHD).

Overall, the cached database has more advantages than the disk-based database because the access time of main memory is always measured in nanoseconds, whereas HDD is measured in milliseconds for the integrated development environment (IDE), small computer system interface (SCSI), or serial advanced technology attachment (SATA). Therefore, there is a difference in the access time of the memory and HHD that can reach 105. The speed difference allows better performance of the main memory than HHD.

The majority of the operations are read in the MIRA analysis application. Read operations take up 97% of the operations performed on MIRA analysis application and the remaining 3% of the operations is on write operations. Most of the focus is paid for the read operations that are crucial in the performance of the system. The results show that read operations in the cache have one advantage over the disk-based database. Figure 4b shows the read operations from the cache that are performed in the first table, namely Preference, which are faster than disk-based table by 11.5 times. For the second table, namely All_Var_Sessions, the read operations from the cache are faster by seven times than the read operations performed in the disk-based table.

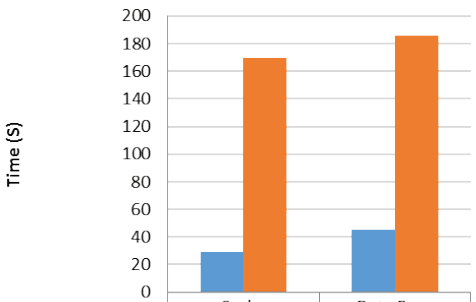


Fig.4 (a). The results for write operation

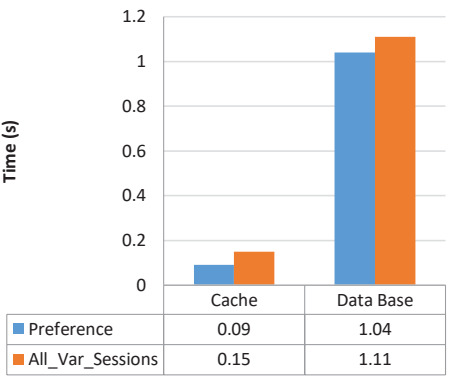


Fig.4 (b). The results for read operation

C. Cloud Service Cost Experiment

A full historical data of the patient's statistical data is required to compare the progress of the patient's performance by the physiotherapist. Therefore, it is recommended to perform the incremental download of the data from the last point of time when evaluating the patient's progress to avoid the increase of monetary cost as the amount of data becomes bigger when the patient plays more games.

This study measured the cost-saving of cloud service in the Perkeso Tun Razak Rehabilitation Center for the download operations of the patients' statistical data from July 1st until August 9th, 2019. The download operation is performed once a day after all the patients completed the required movements and games. Table 3 shows the details and cost calculation with numerical results for the optimised cost of the downloaded data.

Table 3 compares two networks with the normal and optimised cost from 1st July until 9th August. The purpose of this measurement is to depict how the proposed framework

reduces the cloud service cost. The table shows that caching the data reduces the cloud service cost. Caching the old data in the server application and performing the incremental download of the newly added data can reduce the monetary cost by 17 times. The incremental download of data reduces the sum of data that is transferred from the cloud to the application server. The result shows that the optimised cost gave a better result in reducing the cost up to 94% compared with the normal cost.

D. Memory Savings

This study experimented and compared the memory space used by the conventional approach and the proposed approach to measure the optimisation achieved in the memory space for the cache by the proposed approach. The experiment used the database of the Perkeso Tun Razak Rehabilitation Centre. The experiment runs two caches for a month on the same dataset by the same number of users before calculating the memory space occupied by each cache.

Figure 5 depicts the saved memory space in our proposed framework and other conventional approaches. The goal of this measurement is to show how the proposed approach is efficient in terms of consuming memory space. As can be observed it reduces the used memory space by 10% which is the required space for internal tracks, keys, and indexed fields for the evicted records. The information is not used after the movement of hot, warm, and cold records in the side of the database by the data manager. Therefore, our proposed scheme outperforms other conventional approaches using the normal database.

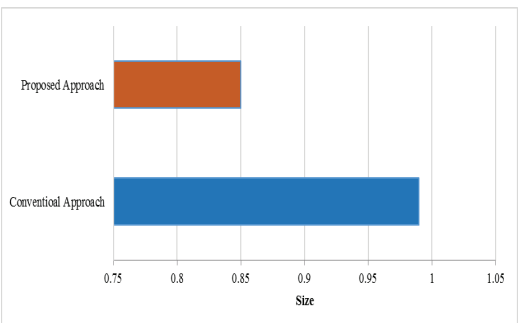


Fig.5. Memory savings

Table 3. Data download details, cost calculation, and cost optimisation results

Date	Transfer/N	X	Cost	Cac	Optim
July	0.02	1	0.02	0	0
July	0.02	23	0.46	22	0.44
July	0.02	23	0.46	0	0
July	0.02	23	0.46	0	0
July	0.02	179	3.58	156	3.12
July	0.02	179	3.58	0	0
July	0.02	179	3.58	0	0
July	0.02	428	8.56	249	4.98
July	0.02	736	14.7	308	6.16
July	0.02	106	21.3	329	6.58
July	0.02	141	28.2	348	6.96
July	0.02	141	28.2	0	0
July	0.02	141	28.2	0	0
July	0.02	170	34.1	294	5.88
July	0.02	203	40.6	326	6.52
July	0.02	243	48.7	404	8.08
July	0.02	285	57.1	421	8.42
July	0.02	285	57.1	0	0
July	0.02	285	57.1	0	0
July	0.02	285	57.1	0	0
July	0.02	325	65.0	395	7.9
July	0.02	361	72.3	366	7.32
July	0.02	394	78.8	321	6.42
July	0.02	427	85.4	332	6.64
July	0.02	455	91.0	280	5.6
July	0.02	455	91.0	0	0
July	0.02	455	91.0	0	0
July	0.02	483	96.6	281	5.62
July	0.02	483	96.6	0	0
July	0.02	508	101.	248	4.96
July	0.02	545	109.	374	7.48
August	0.02	578	115.	330	6.6
August	0.02	578	115.	0	0
August	0.02	578	115.	0	0
August	0.02	600	120.	219	4.38
August	0.02	635	127.	348	6.96
A	0.02	670	134.	354	7.08
August	0.02	692	138.	220	4.4
August	0.02	725	145.	327	6.54
August	0.02	771	154.	466	9.32
Totals		131	263	771	154.3
x factor optimisation		17.10			

VI. DISCUSSION

Remote medicine, specifically telerehabilitation, has been rapidly expanding as an alternative or a complement to conventional face-to-face physical therapy since its development at the turn of the 21st century [44]. Telerehabilitation therapy enables the clinicians and rehabilitation specialists to monitor the patient's at-home recovery frequently in order to intervene earlier [45-47]. MIRA is one of the most popular telerehabilitation systems [48], which provides meaningful feedback such as statistical data about the patients' movements from the devices equipped with motion tracking sensors [31-33].

In MIRA platform, the statistical data of the patients' movement in the temporary storage and then is synchronised to the storage service of online cloud data [49]. So, users can retrieve their data anytime and anywhere from every device. Also, they can take care of themselves via these systems according to their health status [12]. However, firstly, this process increases the main memory usage; moreover, it exposes extra financial cost to the user due to using cloud service. In addition, reading the data from the cloud via the internet servers causes latency problem. To solve this problem, cache memory is employed in this paper. Main memory database and cache employment are two popular solutions that use internal tracking in the main memory to keep track of the records that are not recently accessed so they can be evicted to the disk [50]. However, in such solutions, the remarkable space of the memory is used for applications with many keys and secondary indexes because of maintaining them all indexed fields of evicted records in the main memory.

To overcome the aforementioned problem, a partitioning technique is presented in this paper. To this end, a custom partitioning is applied to the database which categorised it into hot/active, warm/inactive, and cold/dead partitions according to the status of the patient. The hot/active partition belongs to active patients; the warm/inactive partition belongs to the patients who are not currently active or have been discharged, and the cold/dead partition belongs to the patient with inactive status; for example, dead patients. Consequently, only the data that belongs to the hot partition were moved to the main

memory. Furthermore, the data of the patients are retained in a different partition in the database according to their new status and the changes are synchronised to the main memory copy. Generally, this partitioning of the database leads to enhance the potential memory space.

Moreover, a cache memory copy of the application server-side is provided for the database that `is stored in the cloud. This cache memory image allows the application to read the data directly from the local memory that overcomes the latency problem that occurred during the reading of data from the cloud hard disks. In addition, utilizing the cache memory image results in minimizing the overall cost of the cloud service by decreasing the number of reading operations from the cloud.

The results revealed our proposed framework is able to produce good quality solutions in terms of utilizing the main memory space and reducing the latency and read operations from the cloud, therefore, reducing the monetary cost. Although the finding of the proposed work is of interest, the data is collected only from the MIRA games for evaluating the proposed framework, which can be considered as a limitation of this work. Therefore, for future work, the performance of the presented scheme needs to be assessed using the data collected from other exergames.

VII. CONCLUSION

The rehabilitation systems face problems in reducing the monetary cost of the whole cloud service and reducing the footprint of the main memory space. On the other hand, users encounter long latency when the required data need to be read from the cloud via the internet and the HDD of the cloud servers. Previous solutions such as the main memory database and cache use internal tracking in the main memory to keep track of the records that are not accessed so they can be evicted to the disk. However, this mechanism leads to reduce the memory space for applications with many keys and secondary indexes due to retaining the keys and all indexed fields of evicted records in the main memory. In this work, we propose an optimal data access framework for telerehabilitation system that divides the database into hot/active, warm/inactive, and cold/dead partitions according to the status of the patient. Moreover, a cache memory image in the application server is provided for the hot partition of the cloud database. Furthermore, the main memory image is created using Apache Ignite which allows the database caching system to cache a huge amount of data like the cache memory database. In addition, it allows the user to choose the data which needs to be cached using the custom partitioning. The main memory copy can reduce the number of reading operations from the cloud when the user requested the data. It also saves the space of the main memory by only caching the hot partition of the database that does not require the internal track of records.

ACKNOWLEDGMENT

The author wishes to send his/her appreciation to the editor and anonymous referees for their constructive comments and criticism. This work is supported by the Newton-Ungku Omar Fund from Malaysia Industry Government Group from High Technology (MIGHT) and code grant USIM/INT-NEWTON/FST/IHRAM/053000/41616.

REFERENCES

[1] Patel, S. and A. Patel, Abig data revolution in health care sector: Opportunities, challenges and technological advancements. International Journal of Information, 2016. 6(1/2): p. 155-162.
[2] Vogt, S., et al., Virtual reality interventions for balance prevention and rehabilitation after musculoskeletal lower limb impairments in young up to middle-aged adults: A comprehensive review on used technology, balance outcome measures and observed effects. International journal of medical informatics, 2019.
[3] Ismail, W., et al. A Conceptual Model of Hybrid Monitoring Rehabilitation Progress of Stroke Patients: A Case Study of a Public Tertiary Hospital in Malaysia. in 2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA). 2019. IEEE.
[4] Maggio, M.G., et al., The Growing Use of Virtual Reality in Cognitive Rehabilitation: Fact, Fake or Vision? A Scoping Review. Journal of the National Medical Association, 2019.
[5] Mehta, D., et al., Digitalization in Health Care Sector: A HR Analytics Perspective.
[6] Deng, Y., Gao, F., & Chen, H. Angle Estimation for Knee Joint Movement Based on PCA-RELM Algorithm. Symmetry, 2020, 12(1), p. 130.
[7] Ayed, I., et al., Vision-Based Serious Games and Virtual Reality Systems for Motor Rehabilitation: A Review Geared Toward a Research Methodology. International journal of medical informatics, 2019.
[8] Valentina, M., et al., Virtual reality in rehabilitation and therapy. Acta Clinica Croatica, 2013. 52(4.): p. 453-457.
[9] Feldman, B., E.M. Martin, and T. Skotnes, Big data in healthcare hype and hope. Dr. Bonnie, 2012. 360: p. 122-125.
[10] Dhayne, H., et al., In Search of Big Medical Data Integration Solutions-A Comprehensive Survey. IEEE Access, 2019. 7: p. 91265-91290.
[11] Raghupathi, W. and V. Raghupathi, Big data analytics in healthcare: promise and potential. Health information science and systems, 2014. 2(1): p. 3.
[12] Goli-Malekabadi, Z., M. Sargolzaei-Javan, and M.K. Akbari, An effective model for store and retrieve big health data in cloud computing. Computer methods and programs in biomedicine, 2016. 132: p. 75-82.
[13] Abadi, D.J., et al. The design of the borealis stream processing engine. in Cidr. 2005.
[14] Reddy, V.K., B.T. Rao, and L. Reddy, Research issues in cloud computing. Global Journal of Computer Science and Technology, 2011.
[15] Khalifa, A., Tele-Rehabilitation Games on the Cloud: A Survey and a Vision. American Journal of Computer Science and Engineering Survey (AJCSES), 2015. 3(2): p. 143-51.
[16] Cai, H., et al., IoT-based big data storage systems in cloud computing: perspectives and challenges. IEEE Internet of Things Journal, 2016. 4(1): p. 75-87.
[17] Mansouri, Y., A.N. Toosi, and R. Buyya, Cost optimization for dynamic replication and migration of data in cloud data centers. IEEE Transactions on Cloud Computing, 2017.
[18] Zhang, Q., et al., CHARM: A cost-efficient multi-cloud data hosting scheme with high availability. IEEE Transactions on Cloud computing, 2015. 3(3): p. 372-386.
[19] Ndikumana, A., et al., Joint communication, computation, caching, and control in big data multi-access edge computing. IEEE Transactions on Mobile Computing, 2019.
[20] Gilheany, S., Ram is 100 thousand times faster than disk for database access. Directions Magazine, 2003.
[21] Park, G.H., Memory system including a nonvolatile memory and a volatile memory, and processing method using the memory system. 2019, Google Patents.
[22] Chen, C.-H., et al., Data Prefetching and Eviction Mechanisms of Cache memory Storage Systems Based on Scheduling for Big Data Processing. IEEE Transactions on Parallel and Distributed Systems, 2019.
[23] Agombar, J.P., et al., Optimizing the management of cache memory. 2018, Google Patents.
[24] Fricker, C., P. Robert, and J. Roberts. A versatile and accurate approximation for LRU cache performance. in 2012 24th International Teletraffic Congress (ITC 24). 2012. IEEE.

[25] Shariff, A.A.M., N. Katuk, and N.H. Zakaria, An overview to pre-fetching techniques for content caching of mobile applications. Journal of Telecommunication, Electronic and Computer Engineering (JTEC),
[26] Gilman, G.R., et al. Challenges and Opportunities of DNN Model Execution Caching. in Proceedings of the Workshop on Distributed Infrastructures for Deep Learning. 2019.
[27] Shim, J., P. Scheuermann, and R. Vingralek, Proxy cache algorithms: Design, implementation, and performance. IEEE Transactions on Knowledge and Data Engineering, 1999. 11(4): p. 549-562.
[28] O'neil, E.J., P.E. O'neil, and G. Weikum, The LRU-K page replacement algorithm for database disk buffering. Acm Sigmod Record, 1993. 22(2): p. 297-306.
[29] Wilson, J.D., et al., Can shoulder range of movement be measured accurately using the Microsoft Kinect sensor plus Medical Interactive Recovery Assistant (MIRA) software? Journal of shoulder and elbow surgery, 2017. 26(12): p. e382-e389.
[30] Moldovan, I., et al. Development of a new scoring system for bilateral upper limb function and performance in children with cerebral palsy using the MIRA interactive video games and the Kinect sensor. in Proceedings of the International Conference on Disability, Virtual Reality and Associated Technologies. 2014.
[31] Moldovan, I., et al. Virtual rehabilitation programme using the MIRA platform, Kinect and Leap Motion sensors in an 81 years old patient with ischemic stroke. in 2017 E-Health and Bioengineering Conference (EHB). 2017. IEEE.
[32] Osgouei, R.H., D. Soulsbv, and F. Bello. An Objective Evaluation Method for Rehabilitation Exergames. in 2018 IEEE Games, Entertainment, Media Conference (GEM). 2018. IEEE.
[33] Yu, X. and S. Xiong, A Dynamic Time Warping Based Algorithm to Evaluate Kinect-Enabled Home-Based Physical Rehabilitation Exercises for Older People. Sensors, 2019. 19(13): p. 2882.
[34] Călin, A., et al., MIRA-UPPER LIMB REHABILITATION SYSTEM USING MICROSOFT KINECT. Studia Universitatis Babes-Bolyai, Informatica, 2011. 56(4).
[35] Azure services platform. 2009 2009-01-23]; Available from: http://www.microsoft.com/azure.
[36] Zukowski, M., et al., MonetDB/X100-A DBMS In The CPU Cache. IEEE Data Eng. Bull., 2005. 28(2): p. 17-22.
[37] Zheludkov, M. and T. Isachenko, High Performance cache memory computing with Apache Ignite. 2017: Lulu. com.
[38] Acharya, S., Apache Ignite Quick Start Guide: Distributed data caching and processing made easy. 2018: Packt Publishing Ltd.
[39] Zhao, T., et al., Cloud-based medical image processing system with anonymous data upload and download. 2013, Google Patents.
[40] Joshi, P.S., Smoothing algorithm for round trip time (RTT) measurements. 2008, Google Patents.
[41] Albeshri, A., C. Boyd, and J.G. Nieto. Geoproof: proofs of geographic loc
[42] ation for cloud computing environment. in 2012 32nd International Conference on Distributed Computing Systems Workshops. 2012. IEEE.
[43] Qiu, M., et al., Energy-aware data allocation with hybrid memory for mobile cloud systems. IEEE Systems Journal, 2014. 11(2): p. 813-822.
[44] Pavlo, A. and M. Aslett, What's really new with NewSQL? ACM Sigmod Record, 2016. 45(2): p. 45-55.
[45] Chughtai, M., et al., The role of virtual rehabilitation in total and unicompartmental knee arthroplasty. The journal of knee surgery, 2019. 32(01): p. 105-110.
[46] Bozic, K.J., et al., Bundled payments in total joint arthroplasty: targeting opportunities for quality improvement and cost reduction. Clinical Orthopaedics and Related Research®, 2014. 472(1): p. 188-193.
[47] Ong, K.L., et al., Prevalence and costs of rehabilitation and physical therapy after primary TJA. The Journal of arthroplasty, 2015. 30(7): p. 1121-1126.
[48] Fu, M.C., et al., Discharge to inpatient facilities after total hip arthroplasty is associated with increased postdischarge morbidity. The Journal of arthroplasty, 2017. 32(9): p. S144-S149. e1.
[49] Cantea, A., et al. Mira. in Proceedings of the 31st International BCS Human Computer Interaction Conference (HCI 2017) 31. 2017.
[50] El Fezazi, M., et al. Knee Functional Telerehabilitation System for Inclusive Smart Cities Based on Assistive IoT Technologies. in The Proceedings of the Third International Conference on Smart City Applications. 2019. Springer.
[51] DeBrabant, J., et al., Anti-caching: A new approach to database management system architecture. Proceedings of the VLDB Endowment, 2013. 6(14): p. 1942-1953.
[52] Jena, S.R. and Z. Ahmad, Response time minimization of different load balancing algorithms in cloud computing environment. International Journal of Computer Applications, 2013. 69(17): p. 22-27.
[53] Li, H.-J. and A.-G. Song (2017). "Architectural design of a cloud robotic system for upper-limb rehabilitation with multimodal interaction." Journal of Computer Science and Technology 32(2): 258-268.
[54] J. Li, S. Zhou and H. Yu, "TAS: A Rehabilitation Training and Administration System Designed for Upper Limb Rehabilitation Robot," 2019 5th International Conference on Control, Automation and Robotics (ICCAR), Beijing, China, 2019, pp. 63-67.
[55] Karime, A., M. Eid, H. Dong, M. Halimi, W. Gueaieb and A. El Saddik (2014). "CAHR: A contextually adaptive home-based rehabilitation framework." IEEE Transactions on Instrumentation and Measurement 64(2): 427-438.
[56] Kato, N., T. Tanaka, S. Sugihara, K. Shimizu and N. Kudo (2016). Trial operation of a cloud service-based three-dimensional virtual reality tele-rehabilitation system for stroke patients. 2016 11th International Conference on Computer Science & Education (ICCSE), IEEE.
[57] Chen, Y. (2015). Real-time point cloud data transmission in Kinect point cloud data based telerehabilitation system.