Optimizing and Enhancing Performance of MVC Architecture based on Data Clustering Technique

Md. Hafizur Rahman
The People`s University of Bangladesh

Faisal Bin Al Abid
International Islamic University Chittagong

M. Naderuzzaman
Dhaka University of Engineering & Technology

Md. Arifur Rahman
The People`s University of Bangladesh

Md. Masud Reza
The People`s University of Bangladesh

ABSTRACT
The frequent use of web based application plays a vital role in our everyday life. MVC (Model View Controller) architecture is used as programmed architectural pattern in order to implement user interfaces. Application software developers utilize MVC (Model View Controller) Architecture for developing web based application. The sizes of databases are increasing day by day in relation with time. Therefore, if we take into account the concept of huge centralized database systems, it has become one of the most challenging criterions for accessing data in acceptable time. Basically, in centralized databases, the records can be classified into two categories considering the access frequency of data. Those records that are being accessed frequently are known as Level 1 data, on the contrary, those accessed in lesser frequency is considered as Level 2 Data. In this paper, we will try to enhance and optimize the performance of MVC architecture based on two parameters namely response time and throughput. The response time and throughput is improved based on the proposed database search algorithm using B+ tree. If the database search engine is idle, the database search engine will look forward to discover whether the intended data is in level 1, otherwise it will search for level 2 data. The level 2 data will be included as level 1 data inside the database or vice versa, for insertion and update operation. However, whether the data is level 1 or level 2 data will be depended upon user choice. Thus, the overall response time as well as throughput will be optimally increased.

Keywords
MVC, Large Database, Database Engine, Access Frequency, Level 1 Data, Level 2 Data.

1. INTRODUCTION
It has been a pretty long time since Relational Database management System (RDBMS) is serving as huge data storage [1]. Each and every year, our newest technology has been improving the collection, storage, and transfer of data at a very low cost.

The model view controller (MVC) is a three different tier architecture design pattern for the dissociation Data Access logic, user interface and control logic. The MVC pattern is more flexible, scalable and improves maintainability and reusability of the application

Smalltalk programming language is implemented for MVC (Model View Controller) architecture. The system is separated by three components: Model expressing domain knowledge, view representing user interface, control that is used to manage the updates to views. MVC has already shown its advantages for interactive applications. It allows multiple representation of the same information, encouraging code utilization, and helping developers to pay attention on a specific application feature [2].

MVC has demonstrated its benefits for interactive applications allowing multiple representations of the same information, promoting the code reutilization, and helping developers to concentrate on a single application aspect [7].

The layers of MVC architecture has been planned to develop web applications specifically throughput and response time. All of these prototypes have different benefits. In this paper, the focus will be shifted to improve the accuracy of throughput and response time.

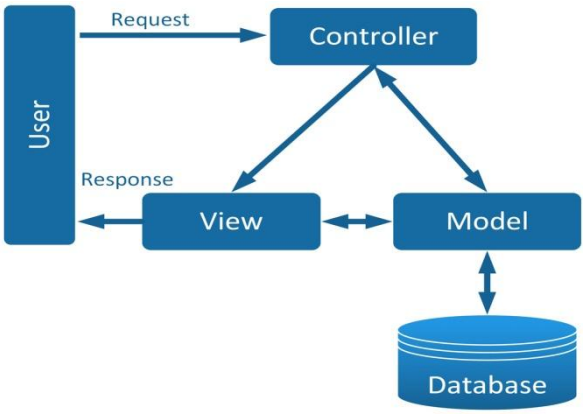


Figure 1: Existing MVC architecture

Table 1: Data access frequency on Level 1 & Level 2 data based on application area.

Application Fields	Time Frame (Days)	Level One Data (%)			Level Two Data (%)		
		INSERT	UPDATE	QUERY	INSERT	UPDATE	QUERY
Education	1450	100	95	90	0	5	10
Sales	180	100	90	85	0	10	15
Finance	360	100	90	80	0	10	20
HR	360	100	90	85	0	10	15
E-Commerce	180	100	85	80	0	15	20
Digital Library & Digital Publishing	720	100	80	65	0	20	35
Banking	360	100	85	70	0	15	30
Credit card Transaction	180	100	90	90	0	10	10
Telecommunication	180	100	85	80	0	15	20
Airlines reservation	180	100	90	90	0	10	10
In Percent		100%	88%	81.50%	0%	12%	18.50%

The percentage of total access on **Level 1 data** can be calculated by the following formula:

$$OL1D_T = 1/N \sum_{i=1}^{N} (IO_i + UO_i + QO_i)/3$$

Where $OL1D_T$ is total operation on **Level 1 data**, and IO, UO and QO is the insert, update and query operation on **Level 1 data** respectively. And N is the number of applications area.

Similarly total operation on **Level 2 data** can be calculated by the following formula:

$$OL2D_T = 1/N \sum_{i=1}^{N} (IO_i + UO_i + QO_i)/3$$

$OL2D_T$ is total operation on **Level 2 data**.

2. RELATED WORK

M. U. Khan at. al. [1] has investigated the design patterns that can improve performance of web applications. They have proposed a new architectural pattern comprising design patters called XWADF for highly scalable and interactive web application development. The design explores possibilities to leverage the performance of web applications through the appropriate use of various design patterns within confines of Model View Controller (MVC).

D. M. Selfa at. al. [2] have introduced different phases of analysis, design and implementation of a database and web application using UML , with the purpose of illustrating a successful application built under MVC. It has a database made up by fifteen relations and a user interface supported by seventeen web pages as central component of the application.

P. Gupta et. al. [3] presented in their paper a web application framework based on MVC in J2EE platform, and extends it with XML so that the framework is more flexible, expansible and easy to maintain. It was a multi tier system including presentation layer, business layer, data persistence layer and database layer. These separate codes, and improve maintainability and reusability of the application.

Adeyinka A. at. al. [4] developed a program incorporates an interactive map which responds to origin and destination selection, by analyzing the relative positions of both locations and creating real-time routes on the road network to display to the user the required path from the origin to the destination and the approximate distance/time required. System design is based on the ModelView-Controller (MVC) design pattern, and the application has been developed using Adobe Flash CS3 (with ActionScript).

Iqbal H. et. al. [5] designed and implemented a Java MVC framework for developing desktop based application which can separate the data, view and control of the software. This application however overcomes the drawbacks of the current application which is existing Java organizes without a structure, which mixes the code of data access, the processing of business logic and graphical user interface (GUI) layer together. As a result the current application creates many problems for software developers, meanwhile; it could not meet the rapid development of application software any more. The new application can help in developing programs.

Patrick L. at. al. [6] presents a combination of the Model-View-Controller pattern with the Evolutionary Acquisition Interdisciplinary Research Project Management for web services development. The intention is to promote an increase in productivity and to facilitate interdisciplinary web services evolution. This consists of an independent database for users' feedback that together with technological opportunities and evolving threats considerations may start a new release of the system allowing a dynamic evolution through acquiring new features or correcting errors, but the new release decision is not automatic.

3. PROPOSED FRAMEWORK

3.1 Basic Idea of Proposed Method

In this paper we are proposing one of the methods of proposed database clustering technique for improving response time and throughput based on MVC pattern. We identified the need for a simple solution for model layer which can reduce the database access time, query cost and improve throughput and response time.

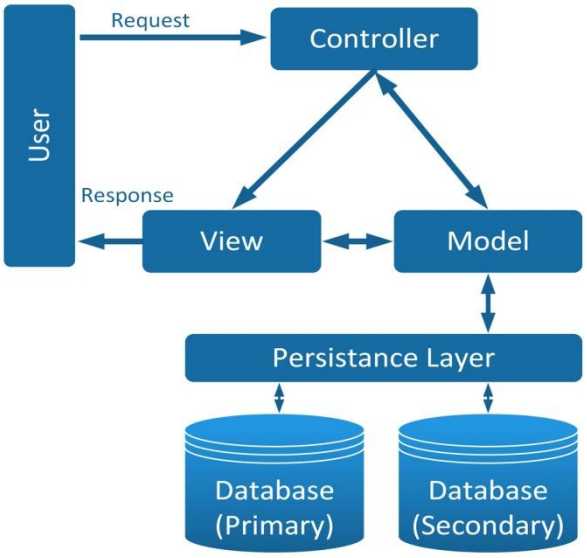


Figure 2: Propose MVC Architecture

3.2 Framework Description

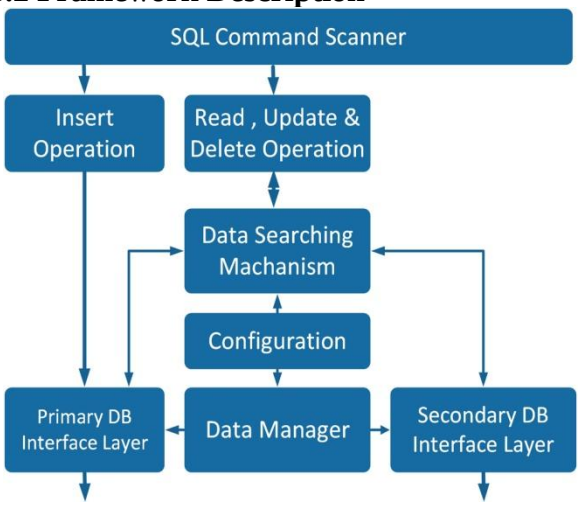


Figure 3: Proposed framework with all its components

3.2.1 SQL Command Scanner

As described above the job of SQL scanner is to identify the type of SQL command it is (insert, update, delete or read operation). For insert it will directly hit on Primary DB and for other operations it will hit to Secondary DB.

3.2.2 Insert Operation:

Insert Operation directly access Primary Database.

3.2.3. Update, Delete, read Operation:

Update, Delete or Read Operation, it will search Primary DB first and if it is not found there, it will certainly be in Secondary DB.

3.2.4. Configuration:

Configuration will have interface with application layer, where the application developer/user will have options to configure how database file will be cluster, the age classification of data etc. This system configuration will also be used for Database connectivity, clustering, data synchronization with others. The configuration includes:

a) Database Server Name or IP Address

b) Database Platform: MS SQL SERVER, MySQL, ORACLE, Others

c) Primary & Secondary Database Name

d) Database User

e) Database Password

f) Database Port (Optional)

g) Application Deployment Date & Time

h) Time Frame in Day

i) Clustering Mechanism Configuration, etc.

3.2.5. Database Manager

Database manager will keep records of existed data on the database files. Any data exceeds the time frame given in the global configuration will be shifted from one database file to another (i.e. Primary DB to Secondary DB or vice versa). Any data or record which was access from the Secondary will be converted as Level 1 data, because according to our classification mechanism any data is accessed recently will be treated as Level 1 Data.

Algorithm:

DataManager(DE_S: Database Engine Status, DS_{PDB}: Data Set of Primary DB, DS_{SDB}: Data Set of Secondary DB, T_P: Time Period)

IF DE_S is idle THEN
BEGIN
TL1D← Get Level 1 Data from DS_{SDB} according to age of data & data access frequency
TL2D← Get Level 2 Data from DS_{PDB} according to age of data & data access frequency
Move TL1D to DS_{PDB}
Move TL2D to DS_{SDB}
END
ELSE
Wait until T_P;

3.2.6. Primary DB Interface Layer & Secondary DB Interface Layer

This layer communicates between database and proposed system.

4. A FRAMEWORK IMPLEMENTATION

Our proposed model has been implemented in programming language by using PHP because increase execution time and reduce system resource & server side open source scripting language. KDE Advanced Text Editor (Kate) has been used to write the source code of the persistence layer. Kate has been chosen because of its comfortable Integrated Development Environments. Kate is developed by C++ Programming Language. Kate has its own file format to maintain the source code. Oracle, MySQL and SQL Server was used as the database. The tools used for our purpose are shown in table 2.

Table 2: System & Application Software used for developing the framework.

System Development Software	Specification
Operating System (Linux Based)	Fedora 17

KDE Advanced Text Editor(Kate)	2.X
Database:	MySQL 5.X
Local Web Server	Apache 2.XX
Programming Language:	PHP 5.X
MVC Architecture:	Codeigniter 3.X

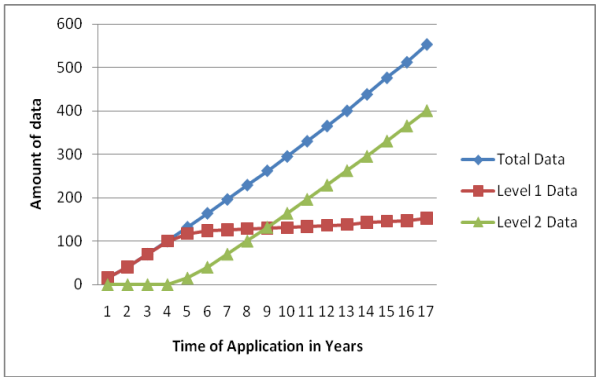
Our proposed model requires and certain minimum hardware. Though we have used much higher configured computers for our purpose. Hardware used for our experiments are shown in table 3.

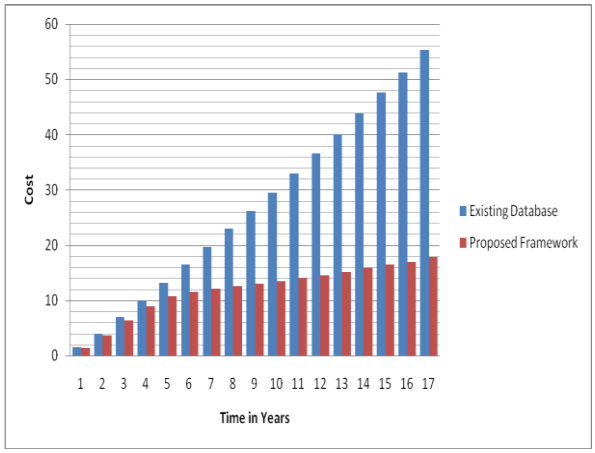
Table 3: Minimum Hardware used for implementing the framework.

Hardware components	Specification
Processor	Intel Dual Core 2.XX GHz (32 bit)
Cache	2048 MB
Primary Memory(RAM)	2048 MB
Secondary Memory(HDD)	80 GB
Chip Set	Intel Chip Set

5. EXPERIMENTAL EVALUATION AND PERFORMANCE ANALYSIS

The functionality offered by our proposed model was for Microsoft's windows operating system. It is observed for the applications mentioned above from the first figure that amount of data is increased for Level 2 data where as the amount of data is pretty much stable for level 1 data in relation with time of application in years. It is clearly seen from the second figure that our proposed method consumes the least cost measured by query time compared with the existing database. To recapitulate, it is absolutely clear that our proposed algorithm outperforms the existing database for the frequently used applications.





6. CONCLUSION AND FUTURE WORK

Idea of splitting an entire database is a novel idea. Append operation is much more common than query operation where as update operation happens almost on the odd occasion. In cases where data are stored and after certain time those data are rarely or not accessed at all for long time, our frame will definitely reduce overhead on database engine by reducing volume of data in the primary database. Hence, it will increase overall response time and throughput of the database. The above experimental results show a certain improvement in performance of our system. In future, we will try to embed such an algorithm that will be feasible for process industry where data change happens very frequently.

7. REFERENCES

[1] M. U. Khan , Dr. T. V. Rao, "XWADF: Architectural Pattern for Improving Performance of Web Applications", IJCSI International Journal of Computer Science Issues, Vol. 11, Issue 2, No 2, March 2014 ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784

[2] Diana M. Selfa, Maya Carrillo, Ma. del Rocío Boone, "A Database and Web Application Based on MVC Architecture", Proceedings of the 16th IEEE International Conference on Electronics, Communications and Computers (CONIELECOMP 2006) 0-7695-2505-9/06 $20.00 © 2006

[3] Praveen Gupta, Prof. M.C. Govil , "MVC Design Pattern for the multi framework distributed applications using XML, spring and struts framework ". (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 04, 2010, 1047-1051

[4] Adeyinka A., Adewaleand Jeremiah, O. Onaolapo, "Development of Web-based Interactive Map Using Object-Oriented Programming Concept". EIE's 2nd Intl' Conf.Comp., Energy, Net., Robotics and Telecom. eieCon2012

[5] Iqbal H. Sarker and K. Apu, "MVC Architecture Driven Design and Implementation of Java Framework for Developing Desktop Application". International Journal of Hybrid Information Technology Vol.7, No.5(2014), pp.317-322. http://dx.doi.org/10.14257/ijhit.2014.7.5.29

[6] Patrick L., Mariwaldo G. Caetano, João Y. Ishihara, David Prata, and George Brito, "Applying MVC to Evolutionary Acquisition IRPM". 2012 International Conference on Information and Knowledge Management (ICIKM 2012) IPCSIT vol.45 (2012) © (2012) IACSIT Press, Singapore

[7] Liu Yong-Jun1, Li Ke-Xi, "Design and Implementation of the New Web Application Framework—JEMSF", IEEE, 2010, pp 190-193.

[8] Md. Umar Khan, T.V. Rao, "Web Application Performance Improvement using new architectural patterns", Proceedings of SARC-ITR International Conference, 2014, Chennai, India

[9] S Rahman, A. M. Ahsan Feroz, Md. Kamruzzaman and M. N. Faruque, "Analyze Database Optimization Techniques", IJCSNS International Journal of Computer Science and Network Security, August 2010, vol. 10, No.8

[10] Bin Li, Jiping Liu, Yi Zhu and Lihong Shi, "Optimization Of Database Capability In The E-Governmental Spatial Aided Decision-Making System", Proceedings of International Symposium on Spatio-temporal Modeling, Spatial Reasoning, Analysis, Data Mining and Data Fusion, Beijing ,China, 2005, vol. XXXVI-2/W25.

[11] P. V. Bommel and Th.P. van der Weide, "Towards Database Optimization by Evolution", In Proceedings of the International Conference on Information Systems and Management of Data, Bangalore, India, July 1992, pp 273 – 287.

[12] Rahman, M.H.; Akter, M.N.; Ahmad, R.B.; Nader-uz-zaman, M.; Rahman, M., "Development of a framework to reduce overhead on database engine through data distribution," in *Electronic Design (ICED), 2014 2nd International Conference on* , vol., no., pp.69-72, 19-21 Aug. 2014

[13] Rahman, M.H; Abid, F.B.A; Zaman, M.N.; Aketer, M.N, "Optimizing and Enhancing Performance of Database Engine using Data Clustering Technique," Proceedings of the 3rd International Conference on Advances in Electrical Engineering, pp.222-225, 17-19 Dec. 2015