An Empirical Study of Performance Improvement of Database Management System using Distributed Processing

Sorapak Pukdesree
School of Science and Technology
Bangkok University
Rama IV rd., Klongtoey, THAILAND
sorapak@gmail.com

Abstract
Modern organizations have stored and managed the information by using database management system software. To apply the proprietary DBMSs Software, organizations have to spend high budget for software’s licenses which depend on the functions and scale of capability to handle their requirements. Furthermore, to implement high performance computer’s capability, organizations have to spend large budget on high-end computer hardware. Alternatively, therefore this research would like to represent the application of applied distributed methodology to DBMSs and computer clusters, which can reduce cost of software licenses and high-end computer hardware significantly.

To implement the distributed methodology, researcher team will use an open source DMBS named MySQL Cluster. This research emphasized on study of scalability performance of distributed database system. The research will apply distributed processing and distributed data on clustering computers. The results of the research represented that distributed methodology can be applied to improve the system’s performance of database system with lower cost of ownership.

Keywords: Clustering Computers, MySQL Cluster Database, Distributed Database, Distributed Processing.

1. Introduction

Computer is a finite automation that has limitation of capacity and performance of each part of the hardware resources and also the software configurations on the system. Even though, the manufacturers of each of part of the hardware resources have keep continue to research and develop more advance hardware technology to improve capacity and performance. The performance of software configurations on the system depends on skills of system administrators and application’s developers. Presently, the computer system’s workload has increased dramatically; there are not only many of computer’s users but also complexity of computer’s software. Those user’s requests and applications commonly access to information which stored on database servers using database management system software. Therefore when there are large numbers of workload from user’s requests, large number of information or large number of access of applications therefore those user’s requests may have to wait for responding for a long period of time depended on type of requests and execution time of each requests.

The number of user’s workload is usually dynamic. We may not know exactly when the workload will be increase or decrease. Some systems may have peak load during working time such as online transactions, however some systems may have peak load in after working time such as batch processing systems. Therefore Information Technology department have to provide sufficiently of capacity of computer resources that can be handle user’s workload as possible as they can. However to provide sufficiently of capacity of computer resources, organizations have to invest large amount of budgets on information management system. Unfortunately most of organizations have constraints on information technology budgets. There may have several alternatively methodologies that can be applied to increase the performance of database system by spending fewer amounts of budgets.

Finally, hybrid approach is the methodology that will apply more than one methodology together to solve the problem. Hybrid approach can obtain the advantages from each methodology even though the system might get more complexity than other approaches. The research paper will apply hybrid approach that combines distributed data methodology [15,19,29,30,31,32,48,51] and distributed
processing methodology on cluster computers [17]. The system architecture design [46] in the research was illustrated in Fig 1.

2. Related work

Wolfgang Emmerich [52] defined the definition of distributed system which consists of a collection of autonomous computers, connected through a network and distribution middleware, which enables computers to coordinate their activities and to share the resources of the system, so that users perceive the system as a single, integrated computing facility.

George Coulouris, Jean Dollimore, Tim Kindberg, and Gordon Blair [19] have proposed that presently distributed systems are everywhere. There are not only typically computers but also the mobile computers and other small computational devices that are attached to the network to construct the small distributed systems. Furthermore, the Internet technology enables users throughout the world to access its services wherever they may be located. Each organization manages an intranet, which provides local services and Internet services for local users and generally provides services to other users in the Internet. They have also proposed that resource sharing is the main motivating factor for constructing distributed systems. Resources such as printers, files, web pages or database records are managed by servers of the appropriate type. For example, web servers manage web pages and other web resources. Resources are accessed by clients – for example, the clients of web servers are generally called browsers.

Hasham Pathan [23] had published his research with titled “MySQL Cluster Database 7: Performance Benchmark”. In the research he was not only benchmark the performance of MySQL Cluster Database 7.0 but also MySQL Cluster Database 6.3. The research illustrated that MySQL Cluster Database 7.0 had significantly higher performance than MySQL Cluster Database 6.3. Highlight of his results are:

- The result of four data storage nodes was 251,000 transactions per minute which was more than four times improvement over the MySQL Cluster 6.3 release.
- The result of two data storage nodes was 143,000 transactions per minute which was more than four times improvement over the MySQL Cluster 6.3 release.
- The architecture of his research in case of two data storage nodes which were running on two Sun Fire X4450 system with eight processor cores per data nodes. The MySQL server nodes were running on a combination of Sun Fire x4450 systems and Sun Fire x4600 systems.
- In case of four data storage nodes, four Sun Fire X4450 system were used to deploy the data nodes which each data node used eight processor cores. The MySQL server nodes were running on a combination of two Sun Fire x4600, one Sun Fire x4240 and four Sun Fire x4450 systems. In both cases all machines were connected using gigabit Ethernet.

Hasham Pathan used DBT2 [43] as the testing benchmark in his research. Database Test 2 (DBT2) is an open source benchmark developed by Open Source Development Labs (OSDL). DBT2 can simulate a typical Online Transaction Processing (OLTP) application that performs transactions with around ten to twenty SQL statements per transactions within five distinct transaction types. The DBT2 benchmark can be used as a good indicator in regards to the expected performance of any applications that perform fairly simple transactions and execute these transactions in a repetitive fashion. To simulate OLTP applications which are typical real-time applications, both DBT2 and MySQL Cluster were configured as an in-memory database. DBT2 can be downloaded from www.iclaustron.com.

The results of his research were very impressive. The architecture of the systems was very high performance machines which were about nine Sun Fire machines. The cost of those machines and other peripherals would be more than one million dollars. This might be suitable for enterprise systems. On the other hand, our research was focusing on the low cost systems such as personnel computers to implement the higher performance distributed database system. Furthermore our research can be used for small and medium of enterprise businesses to make plans of database workload to meet their business’s requirements.

Javier Bilbao and team [24] had proposed the paper with title “Easy clustering with
openMosix”. The paper presented three ways to implement a cluster with openMosix as software to balance the nodes and to migrate the processes: clusterKnoppix, Chaos and the use of diskless. The result of some evaluations shown that the use of a cluster required lower execution time, but that it is conditioned by some external constraints such as the traffic of the network, etc.

Guangzhong Sun and team [21] had proposed the paper with title “A Framework for parallel Genetic Algorithms on personnel Cluster”. The paper presented FPGA-Cluster made use of parallel programming and island model to implement parallel genetic algorithm. With the help of FPGA-Cluster, programmers can turn genetic algorithms into parallel algorithms intuitively, rapidly and effectively. FPGA -Cluster can also help the design and analysis of parallel genetic algorithms.

Amany Abd Elsamea and team [4] had proposed the paper with title “PC cluster as a platform for parallel applications”. They built a Linux cluster using a number of relatively of cheap computers which connected together by a network to produce widespread, efficient sharing of resources. This paper approved that the cluster is an efficient platform for running heavy computation applications. Also, it is clear that the cluster gives better price to performance ratios. As a future work, this cluster will be extended over WAN to form a Grid which provides flexible, secure, coordinated resource sharing.

Anon Sukstrienwong [8] had proposed the paper with title ”Forming Buyer Coalition with Bundles of Items by Ant Colony Optimization”. The paper proposed method called ACBC is presented to form a buyer coalition with bundle of items by using the ant colony optimization technique. The algorithm is based on an imitation of the foraging behavior of real ants. The ants construct the trail by depositing pheromone after moving through a path and updating pheromone value associated with good or promising solutions thought the lines of the path. The experimental results show that in most cases the average of total utility of any coalitions formed by ACBC is better than the average of total utility derived by Group Package String scheme. In future, we plan to adapt the proposed algorithm to other real complex world problems to see how well to apply.

Anon Sukstrienwong [9] had proposed the paper with title ” Genetic Algorithms for Multi-Objectives Problems under Its Objective Boundary”. This paper presents the new GA-based algorithm called GAsOB of solving the particular multi objective problems in which some objectives are requested to be in certain intervals. From the experiment, there are two factors involved in efficiency of GAsOB scheme; immigration rate, number of eras. The resulting showed that the best immigration rate for the GAsOB scheme is approximately 5%. And, GAsOB scheme needs at least 5 eras to get the best results. The experiment shows that the proposed algorithm is able to find the solution about 98% of the accuracy value within a very small amount of time. For future works, the proposed algorithm will investigate in real-world problems and compare with other well-known approaches.

Our previous research paper, we had presented his previous research which had implemented distributed database System using MySQL Cluster 5.0 [1,3,16,20,22] the previous version on older personnel computers. Those machines had specifications with a single core on single processor chip Pentium 4 1.8 GHz processor, 256 MB SDRAM single channel, 20 GB-IDE HDD and one on-board 100 Mbps fast Ethernet NIC. This research applied only distributed database [49]. The maximum number of data storage nodes was eight storage nodes. We used SQL-Bench as benchmark tool. The best result was 361 seconds with 20.31% improvement. The performance of the old evaluation was greatly lower than the new evaluation. That might cause of lower computer’s capacity, older version of operating system, older engine of MySQL Cluster and lower network bandwidth.

Our previous research paper, we also had presented another research with new and higher specifications of personnel computers. The specifications of each machines were two cores on single chip Intel E6400 processor, 2GB DDR-RAM single channel, 250GB SATA-II HDD and one on-board 1 Gbps Ethernet network. The evaluation had worked on MySQL Cluster 7.0 and Red Hat Enterprise Linux 5.4. This research applied only distributed database. The maximum
number of data storage nodes was eight storage nodes. He used SysBench as benchmark tool. The best result was 177,816 transactions within 300 seconds with 20.31% improvement using four storage nodes. Furthermore he also evaluated the system using Benchw benchmark [11] tool. The best result was 7.807 seconds with 54.60% improvement using eight storage nodes.

3. Hardware Implementation

In this research, we had used ten personal computers that were in the computer laboratory as illustrated in Fig 1. The specifications of each computer were a single chip Intel Core 2 Duo E6400, 2 GB-667 MHz of RAM, 250 GB-SATA II of hard drive and on boarded 1000 Mbps of network interface. Presently, each computer was cost approximately less than six hundred dollars. Therefore the total cost of the entire system would approximately six thousands and five hundreds dollar. All of computers were connected using two eight-port 1000 Mbps switching hubs with CAT5e [13] wired cable connections. These computers were configured as Cluster computers [47,50] using personnel computers.

The system was a closed system that prevented other factors that might affect the results of the experimental. Even though these computers did not have high performance as new personnel computers or high performance servers but there were several advantages such as significantly lower cost of hardware and software licenses also the simplicity to configuration of personnel computers. Furthermore the personnel clustering could be scaled up the capability depended on the required capacity. Fortunately we did not have to spend budgets on personnel computers due to the machines are in computer laboratory within the department. We only spent on two eight-port 1000 Mbps switching hubs because the current switching hubs in this computer laboratory are still only 100 Mbps.

In the distributed data approach, the network bandwidth is one of the very importance factors that will affect the results of the testing experimental. Even though presently the 1000 Mbps is standard and widely add-on in most personnel computers but the wired cable is only CAT5e that has maximum 350 Mbps of network capacity. That means we may not utilize the full capacity of 1000 Mbps connections because of the limitation of capability of CAT5 wired cable. In the next research we may use CAT6 [14], CAT6e or CAT7 that has 550 Mbps or 1000Mbps to improve the network capacity. Unfortunately we do not have planned to use very high speed networks as Myrinet, Infiniband or 10 Gbps-Ethernet network systems.

![System architecture design in this research.](image)

4. Software Implementation

We decided to use Red Hat Enterprise Linux 5 [34] as our operating system for the system. RHEL5 is one of operating system that has reliable, potential, stable, secure and so forth. We did not test the system on others open source operating system such as fedora or FreeBSD. In our assumption, we are going to deploy the system which we do not want to get risk or unreliable situations. Therefore we should not deploy unstable operating system. RHEL5 also provides support for their customers via...
subscription and also the system can be updated patches or packages via internet that can be very useful for system administrator to fix or upgrade the system software.

Red Hat is one of the world's leading open source technology solutions provider which under GNU General Public License. Therefore we did not have to spend for the software license as proprietary operating system. We may optionally have to pay for subscription for update or fix the system when needed.

Previously we had researched and discovered the MySQL Cluster 5.0 [37,38] for the past four years. Therefore we have some experiences and familiar with MySQL Cluster.

In this research we use MySQL Cluster 7.0 as our distributed DBMS which is the latest version at this time. MySQL Cluster 7.0 provides many advance features as enterprise DBMSs such as HA or online duplication of database. We installed only required packages on each type of MySQL Cluster components, for example management node, SQL nodes and storage nodes. MySQL Cluster supports both disk-based and in-memory database. In this research we use in-memory database approach because this type provides greatly high responsiveness than disk-based approach. The access time of in-memory database is significantly faster than disk-based approach. MySQL Cluster 7.0 also supports up to eight threads of data nodes in parallel [18,45] that is very suitable for present processors multi-thread or multi-core era.

There are some researches papers represented that the new version of MySQL Cluster 7.0 has greatly high performance than previous versions. Before the evaluation the database performance, we had configured tuning the operating system and also MySQL Cluster Database [25,27,28,44] to perform the highest effective performance in the environment.

5. System Evaluation

To evaluate the performance [29,32,33,35,36,39,42] of distributed database, we had used SysBench [2,3] as our benchmark in this research. There are some other benchmark tools including proprietary or open source tools. Even though there are many of open source benchmark tools but some of them were not work in our environment that may cause of our mistaken of the system configurations. There are also many of commercial benchmark tools either but we had chosen SysBench. SysBench is a database benchmark tool developed by MySQL that supports both common and distributed database. SysBench is a modular, cross-platform and multi-threaded benchmark tool for evaluating OPERATING SYSTEM parameters that are important for a system running the database under intensive load. Primarily written for MySQL server benchmarking, SysBench will be further extended to support multiple database backend, distributed benchmarks and third-party plug-in modules. The operations within SysBench include alter-table, large table, connect, create, insert, select and transactions. In this research, we have customized some part of scripts of SysBench to support MySQL Cluster when created the table on the database. We have changed default database engine from MyISAM to NDBCLUSTER to utilize the distributed data methodology approach.

We started the test focus on the term of the number of processed transactions within the specific of time 300 seconds. Furthermore we also have used four storage nodes to demonstrate the distributed database methodology. In our previous research paper, we had discovered that four storage nodes given the best result of test in the specific environment. The four storage nodes have been configured using eight threads due to each processor has two processing cores that was suitable in the environment. Furthermore MYSQL Cluster supports up to eight processing threads. The results of SysBench testing with read/write operations were illustrated in table 1. The result of one processing nodes was 158,285 transactions. The result of two processing nodes was 310,238 transactions which can improve the performance to 96.00%. The result of three processing nodes was 438,727 transactions which can improve the performance to 177.18%. The result of four processing nodes was 494,995 transactions which can improve the performance to 212.72%. The result of five processing nodes was 589,873 transactions which can improve the performance to 272.67%. The result of six processing nodes was 601,802 transactions which can improve the performance to 259.01%. The result of seven processing nodes was 568,252 transactions which can improve the performance to 259.01%.
The result of eight processing nodes was 590,565 transactions which can improve the performance to 273.10%.

The results were illustrated that when we increased more processing nodes therefore the number of succeed transactions trended to grow up as illustrated in Fig 2. The best performance ratio of this test was 280.20 % by using six processing nodes. In case of eight processing nodes, the performance was slightly downgrade than six processing nodes. We had analyzed that we have limited not only specifications of computer hardware and also limited network capacity.

Table 1 also illustrated the average number of transactions per second that can be executed successfully within a specific of time. By using one processing nodes, the average transaction was only 527 transactions. By using two processing nodes, the average transaction was 1,034 transactions. By using three processing nodes, the average transaction was 1,465 transactions. By using four processing nodes, the average transaction was 1,649 transactions. By using five processing nodes, the average transaction was 1,966 transactions. By using six processing nodes, the average transaction was 2,006 transactions. By using seven processing nodes, the average transaction was 1,894 transactions. Finally by using eight processing nodes, the average transaction was 1,968 transactions. The results of average number of transactions per second were corresponding with the results of the number of transactions with a specific of time.

Table 1. SysBench Testing Result (Read/Write Operations)

<table>
<thead>
<tr>
<th>Processing nodes</th>
<th>Total Transactions</th>
<th>Performance Ratio</th>
<th>Average Transactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>158,285.00</td>
<td></td>
<td>527.62</td>
</tr>
<tr>
<td>2</td>
<td>310,238.00</td>
<td>+96.00 %</td>
<td>1,034.13</td>
</tr>
<tr>
<td>3</td>
<td>438,727.00</td>
<td>+177.18 %</td>
<td>1,462.42</td>
</tr>
<tr>
<td>4</td>
<td>494,995.00</td>
<td>+212.72 %</td>
<td>1,649.98</td>
</tr>
<tr>
<td>5</td>
<td>589,873.00</td>
<td>+272.67 %</td>
<td>1,966.24</td>
</tr>
<tr>
<td>6</td>
<td>601,802.00</td>
<td>+280.20 %</td>
<td>2,006.01</td>
</tr>
<tr>
<td>7</td>
<td>568,252.00</td>
<td>+259.01 %</td>
<td>1,894.17</td>
</tr>
<tr>
<td>8</td>
<td>590,565.00</td>
<td>+273.10 %</td>
<td>1,968.55</td>
</tr>
</tbody>
</table>

Figure 2. Testing Result (Read/Write Operations)

The results of SysBench Testing with read only operations were illustrated in table 2. The result of one processing nodes was 195,819 transactions. The result of two processing nodes was 381,084 transactions which can improve the performance to 94.61%. The result of three processing nodes was 527,803 transactions which can improve the performance to 169.54%. The result of four processing nodes was 576,479 transactions which can improve the
performance to 194.39%. The result of five processing nodes was 639,666 transactions which can improve the performance to 226.66%. The result of six processing nodes was 692,752 transactions which can improve the performance to 253.77%. The result of seven processing nodes was 666,140 transactions which can improve the performance to 240.18%. The result of eight processing nodes was 667,896 transactions which can improve the performance to 241.08%.

The results were illustrated that when we increased more processing nodes therefore the number of succeed transactions trended to grow up as illustrated in Fig 3. The best performance ratio of this test was 253.77 % by using six processing nodes. In case of eight processing nodes, the performance was slightly downgrade than six processing nodes. We had analyzed that we have limited not only specifications of computer hardware and also limited network capacity.

Table 2 also illustrated the average number of transactions per second that can be executed successfully within a specific of time. By using one processing nodes, the average transaction was only 652 transactions. By using two processing nodes, the average transaction was 1,270 transactions. By using three processing nodes, the average transaction was 1,759 transactions. By using four processing nodes, the average transaction was 1,921 transactions. By using five processing nodes, the average transaction was 2,132 transactions. By using six processing nodes, the average transaction was 2,309 transactions. By using seven processing nodes, the average transaction was 2,220 transactions. Finally by using eight processing nodes, the average transaction was 2,226 transactions. The results of average number of transactions per second were also corresponding with the results of the number of transactions with a specific of time.

Table 2. SysBench Testing Result (Read Only Operations) focused on number of Transactions.

<table>
<thead>
<tr>
<th>Storage nodes</th>
<th>Total Transactions</th>
<th>Performance Ratio</th>
<th>Average Transactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>195,819.00</td>
<td></td>
<td>652.73</td>
</tr>
<tr>
<td>2</td>
<td>381,084.00</td>
<td>+94.61 %</td>
<td>1,270.28</td>
</tr>
<tr>
<td>3</td>
<td>527,803.00</td>
<td>+169.54 %</td>
<td>1,759.34</td>
</tr>
<tr>
<td>4</td>
<td>576,479.00</td>
<td>+194.39 %</td>
<td>1,921.60</td>
</tr>
<tr>
<td>5</td>
<td>639,666.00</td>
<td>+226.66 %</td>
<td>2,132.22</td>
</tr>
<tr>
<td>6</td>
<td>692,752.00</td>
<td>+253.77 %</td>
<td>2,309.17</td>
</tr>
<tr>
<td>7</td>
<td>666,140.00</td>
<td>+240.18 %</td>
<td>2,220.47</td>
</tr>
<tr>
<td>8</td>
<td>667,896.00</td>
<td>+241.08 %</td>
<td>2,226.32</td>
</tr>
</tbody>
</table>

Figure 3. SysBench Testing Result (Read Only Operations)

Next we focused the test in the term of the number of processed requests in the specific of time. The results of SysBench testing with read/write requests were illustrated in table 3. The result of one processing nodes was 3,017,439 requests. The result of two processing nodes was 5,914,169 requests.
which can improve the performance to 96.00%. The result of three processing nodes was 8,363,597 requests which can improve the performance to 177.18%. The result of four processing nodes was 9,436,252 requests which can improve the performance to 212.72%. The result of five processing nodes was 11,244,942 requests which can improve the performance to 272.67%. The result of six processing nodes was 11,472,348 requests which can improve the performance to 280.20%. The result of seven processing nodes was 10,832,773 requests which can improve the performance to 259.01%. The result of one processing nodes was 11,253,473 requests which can improve the performance to 272.95%.

The results were illustrated that when we increased more processing nodes therefore the number of succeed transactions trended to grow up as illustrated in Fig 4. The best performance ratio of this test was 280.20% by using six processing nodes. In case of eight processing nodes, the performance was slightly downgrade than six processing nodes. We had analyzed that we have limited not only specifications of computer hardware and also limited network capacity.

Table 3 also illustrated the average number of requests per second that can be executed successfully within a specific of time. By using one processing node, the average request was only 10,058 requests. By using two processing nodes, the average request was 19,713 requests. By using three processing nodes, the average request was 27,878 transactions. By using four processing nodes, the average request was 31,454. By using five processing nodes, the average request was 37,483 requests. By using six processing nodes, the average request was 38,241 requests. By using seven processing nodes, the average request was 36,109 requests. Finally by using eight processing nodes, the average request was 37,511 requests. The results of average number of requests per second were corresponding with the results of the number of requests with a specific of time.

Table 3. SysBench Testing Result (Read/Write Operations) focused on number of Requests within 300 seconds.

<table>
<thead>
<tr>
<th>Processing nodes</th>
<th>Total Requests</th>
<th>Performance Ratio</th>
<th>Average Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3,017,439.16</td>
<td>+96.00%</td>
<td>10,058.13</td>
</tr>
<tr>
<td>2</td>
<td>5,914,169.14</td>
<td>+177.18%</td>
<td>19,713.90</td>
</tr>
<tr>
<td>3</td>
<td>8,363,597.22</td>
<td>+212.72%</td>
<td>27,878.66</td>
</tr>
<tr>
<td>4</td>
<td>9,436,252.27</td>
<td>+272.67%</td>
<td>31,454.17</td>
</tr>
<tr>
<td>5</td>
<td>11,244,942.31</td>
<td>+280.20%</td>
<td>37,483.14</td>
</tr>
<tr>
<td>6</td>
<td>11,472,348.58</td>
<td>+259.01%</td>
<td>38,241.16</td>
</tr>
<tr>
<td>7</td>
<td>10,832,773.83</td>
<td>+272.95%</td>
<td>36,109.25</td>
</tr>
<tr>
<td>8</td>
<td>11,253,473.82</td>
<td>+272.95%</td>
<td>37,511.58</td>
</tr>
</tbody>
</table>

Figure 4. SysBench Testing Result (Read/Write Operations) focused on number of Requests within 300 seconds.

Finally we focused the test in the term of the number of processed requests in the specific of time. The results of SysBench testing with read-only requests were illustrated in table 4. The result of one processing nodes was 2,750,630 requests. The result of two processing nodes was 5,352,958 requests...
which can improve the performance to 94.61%. The result of three processing nodes was 7,413,871 requests which can improve the performance to 169.54%. The result of four processing nodes was 8,097,606 requests which can improve the performance to 194.39%. The result of five processing nodes was 8,985,172 requests which can improve the performance to 226.66%. The result of six processing nodes was 9,730,853 requests which can improve the performance to 253.77%. The result of seven processing nodes was 9,357,043 requests which can improve the performance to 240.18%. The result of eight processing nodes was 9,377,827 requests which can improve the performance to 240.94%.

The results were illustrated that when we increased more processing nodes therefore the number of succeed transactions trended to grow up as illustrated in Fig 5. The best performance ratio of this test was 253.77 % by using six processing nodes. In case of eight processing nodes, the performance was slightly downgrade than six processing nodes. We had analyzed that we have limited not only specifications of computer hardware and also limited network capacity.

Table 4 also illustrated the average number of requests per second that can be executed successfully within a specific of time. By using one processing nodes, the average request was only 9,168.68 requests. By using two processing nodes, the average request was 17,843 requests. By using three processing nodes, the average request was 24,712 requests. By using four processing nodes, the average request was 26,992 requests. By using five processing nodes, the average request was 29,950 requests. By using six processing nodes, the average request was 32,436 requests. By using seven processing nodes, the average request was 31,190 requests. Finally by using eight processing nodes, the average request was 31,259 requests. The results of average number of requests per second were corresponding with the results of the number of requests with a specific of time.

<table>
<thead>
<tr>
<th>Processing nodes</th>
<th>Total Requests</th>
<th>Performance Ratio</th>
<th>Average Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,750,603.60</td>
<td>+94.61%</td>
<td>9,168.68</td>
</tr>
<tr>
<td>2</td>
<td>5,352,958.89</td>
<td>+169.54%</td>
<td>17,843.20</td>
</tr>
<tr>
<td>3</td>
<td>7,413,871.21</td>
<td>+194.39%</td>
<td>24,712.90</td>
</tr>
<tr>
<td>4</td>
<td>8,097,606.50</td>
<td>+226.66%</td>
<td>26,992.02</td>
</tr>
<tr>
<td>5</td>
<td>8,985,172.95</td>
<td>+253.77%</td>
<td>29,950.58</td>
</tr>
<tr>
<td>6</td>
<td>9,730,853.83</td>
<td>+240.18%</td>
<td>32,436.18</td>
</tr>
<tr>
<td>7</td>
<td>9,357,043.73</td>
<td>+240.94%</td>
<td>31,190.15</td>
</tr>
<tr>
<td>8</td>
<td>9,377,827.54</td>
<td></td>
<td>31,259.43</td>
</tr>
</tbody>
</table>

Table 4. SysBench Testing Result (Read Only Operations) focused on number of Requests.

Figure 5. SysBench Testing Result (Read Only Operations) focused on number of Requests within 300 seconds.
6. Conclusion

Wherever Times New Roman is specified, Times Roman, or Times may be used. If neither is available on your word processor, please use the font closest in appearance to Times New Roman that you have access to. Please avoid using bit-mapped fonts if possible. True-Type 1 fonts are preferred.

The main objective of the research was to evaluate the distributed database approach that can improve the performance of database system. Instead of apply enterprise database management systems, this research used an open source DBMS named MySQL Cluster. And also the research had used personnel computers within computer laboratory. Therefore the research team can reduce cost of software licenses and hardware implementations drastically.

To implement distributed data methodology, we had used six storage nodes. In our previous research, we had evaluated that six storage nodes will produce the highest performance in our testing environment. Therefore we had chosen the highest performance of the distributed data storage for the distributed database processing.

We evaluated the distributed database processing using SysBench benchmark tool. We had tested two types of operations as read/write and read-only. The results illustrated that when we increased the number of processing nodes which processed database’s query, the number of succeed transactions trended to improve gratefuly and to improve the average number of succeed transactions per second. The best case of the results was +280.20% improvements with using six processing nodes.

The results of the testing represented that the distributed data methodology and distributed processing can be applied to improve the processing of database’s query. Furthermore the system can be scalable performance to meet the requirements of ever-changing market conditions and stringent service-level agreements.

The evaluation may be limited by the maximum number of processing nodes to eight data storage machines. If we have some opportunities, we will try to evaluate more data storage nodes and more processing nodes. We may also try to improve some other factors that would affect the system performance. Furthermore, from the same system design we have plan to change from CAT5e [13] wired connection to CAT6 [14], CAT6e or CAT7 (if available in Thailand) wired connection to improve the network capacity that may affect the system improvement. We also plan to use more benchmark tools such as DBT2 to evaluate more aspects of the system performance.

7. Acknowledgment

This research was supported by Bangkok University who has supported some facilities, budgets and computers in the computer laboratory for the research and also supported the previous research papers. Furthermore we would like to appreciate 2b-cert Company which supported two eight-port 1000 Mb switching hubs, CAT5e and CAT6 wired cables and knowledge from expert faculties. Finally we would like to appreciate Mr. Vitalwonhyo Lacharoj and Mr. Parinya sirisang who helped the evaluation of the system performance.

10. References


[34] Mary Edie Meredith, Duc Vianney, Linux 2.6 Performance in the Corporate Data Center, Open Source Development Labs (OSDL).


43

An Empirical Study of Performance Improvement of Database Management System using Distributed Processing
Sorapak Pukdesree