Research on Performance Optimization of Web Application System based on JAVA EE

Zengyu Cai, Jingxiao Li, Jianwei Zhang

1School of Computer and Communication Engineering, Zhengzhou University of Light Industry Henan Key Laboratory of Food Safety Data Intelligence, Zhengzhou
2Software Engineering College, Zhengzhou University of Light Industry Henan Key Laboratory of Food Safety Data Intelligence, Zhengzhou

ABSTRACT. With the development of informatization, the Web application system is being used by more and more enterprises and institutions. With the increase of the user group, the concurrent visits are increasing, which leads to the poor service quality of Web system. Several high-concurrency solutions existed, but they address only one aspect of the performance problem, and small and medium enterprises need to consider the cost. This paper will analyze the performance factors affecting the Web application system through multiple levels, aiming to build a Web application system with high availability, high concurrency, high expansion, and less cost. Based on the factors affecting the performance of the Web application system, the project of the online mall system was optimized, and the service support of the Web application system was adjusted to make it more suitable for the system requirements. Besides, related SQL statements were optimized, Java code layer was optimized, user experience layer was optimized, and performance testing and analysis were completed.

1. INTRODUCTION
As technology and society progress, people's lifestyle changes gradually, and more and more people use web-based application systems. Accessing system services is getting slower and slower, and even when there's no response to a crash, the high-concurrency problem begins to be noticed and attract attention gradually. In just a few years, there have been several solutions to the problem of high concurrency in a few years [1-2]. By considering the problem of server horizontal expansion, LiveJournal company in foreign countries put forward a high-concurrency Web server technology. In the case of high concurrency problem, a hardware upgrade of a physical machine tends to yield better benefits. However, the processing power of a single physical machine is limited, LiveJournal also proposes a division technology, which expands the pressure database according to the business demand [3]. With the development of distributed and cloud computing, the service capacity of the Web application system is becoming stronger. Relying on distributed services and cloud computing platform, the performance problem of Web application system has been solved, but there are still many problems. While cloud servers are relatively cheap these days, cloud computing services are relatively expensive. Older frameworks such as the SSH framework and single-server deployments are still being adopted by most SMEs. Therefore, considering the cost of upgrading and maintenance, the
application system of small and medium-sized enterprises has not achieved high concurrency. In this paper, through the optimization of the online mall performance to provide optimization scheme reference for small and medium-sized websites, aiming to use less cost to build high availability, high concurrency, high expansion of the Web application system.

2. **Analysis of factors affecting web application system performance**

When it comes to the high concurrency Web application system, the first thing that comes to mind is the large e-commerce Web application system such as Taobao and Jingdong. During the annual "double eleven", the number of visits to the server is huge, and the massive database operation is likely to cause problems such as goods cannot be traded, slow page response or even crash, which reduces the user experience and the operation effect of the platform.

Therefore, the problems that may be encountered by the system during operation should be analyzed at the outset of the system design, and be circumvented by appropriate technology and be easily extended and maintained [4]. So it is crucial to analyze the factors that affect the performance of the Web application system. When analyzing the factors that affect the performance of the Web application system, it should be analyzed from the following aspects:

2.1 **User experience layer**

User experience is the feeling brought to users by the response time when users open the Web application system, which is affected by the problems such as system throughput, front-end script error rate, asynchronous request error rate, static resource 404, etc. When the above problems exist in the requested resource, the request will time out, resulting in slow overall response and poor user experience. Therefore, it is necessary to analyze whether the above problems exist in the system.

2.2 **The network layer**

The size of the server network bandwidth determines the response speed of the Web application system. Theoretically, static resources with high exit bandwidth will load faster, and the network transmission speed can generally be tested with the ping command.

2.3 **Server layer**

The number of requests carried by the server is limited, and the server load state also determines the service efficiency of the system. Therefore, the server should be analyzed to see if there are an aspect and a configuration aspect of the server.

2.4 **Code layer**

The speed of the system is also determined by the quality of the code. If the system has multiple queries like multiple nested loop queries, it will show a slow response or even crash in the user experience layer. Therefore, attention should be paid to the performance of the Java language in development and optimization.

2.5 **Data storage layer**

In the high concurrency Web application system, the database concurrency problem is a difficult problem to solve. It is constrained by SQL statement, locking mechanisms, and concurrency, so in the system database performance analysis, we should focus on the SQL and frequent database operation and concurrent "dirty read" problem.

2.6 **Physical hardware layer**

This layer is mainly reflected by the system business processing, which is affected by CPU and memory.
3. Performance optimization scheme of the web application system based on online mall system

In order to prove the feasibility, this paper carries out optimization design, analysis and comparison according to the online shopping mall system developed based on Java language, proposes the corresponding theoretical solution, and realizes it in the system step by step, and finally completes the performance optimization of the application system through the corresponding comparison test.

3.1 Online mall system module design

3.1.1 System Function Design

In the traditional mall website, the main purpose of users to visit the website is to choose and buy commodities, so in the online mall design, there should be commodity display, commodity classification, shopping cart, order, payment and other functions. In order to complete all the processes of shopping online, the functions of online shopping mall system can be roughly divided into the following six modules: rotation chart module, user module, commodity display module, shopping cart module, order module, and payment module, See figure 1.

3.1.2 Database Design

To meet the functional requirements of the online mall system, Six database tables are designed in the online mall, which are rotation chart(ad), classification navigation menu table(cat_item), comment table(comment), commodity list(goods_item), order table(shop_orders) and user table(user).

3.2 Analysis of system performance problems

3.2.1 User experience layer problem analysis

The home page of the online mall system is the most requested page of the user. In the home page, the browser needs to load a lot of data, such as: obtain static HTML page, js and CSS, obtain user login information, obtain the data of the left navigation bar classification, obtain the rotation chart, obtain the list of page items and obtain the number of shopping cart items, etc.

The data of the left navigation bar of classification, the data of the rotation chart, and the product information on the page should be read and displayed from the database. A large number of reading operations will lead to problems such as database pressure and server bandwidth occupation. It should be possible to avoid that frequent query database operation and to distribute the single server pressure as much as possible in the system optimization to reduce the server response time and improve the user experience.

3.2.2 Server layer problem analysis

The Web server used in this article is a free, open-source tomcat server. By default, the JVM heap memory size and the maximum number of connections requested for a given set of configurations in the tomcat configuration file are low. So there is more room for optimization in the configuration.

3.2.3 Data storage layer problem analysis

Problems include oversold items, frequent operation of orders, unimportant data storage, and database concurrency.
3.2.4 Code layer problem analysis
Java provides many high concurrency development interfaces and classes. Therefore, the development and optimization should not only focus on the performance of business logic but also pay attention to the use of related interfaces and methods and related development specifications. Also, the rational use of caching technology in code fragments can also bring benefits to system performance.

3.3 System optimization scheme
Among the factors affecting the performance of the Web application system, the network layer and physical hardware layer of the online mall system can be solved by renting or building high-quality servers [5], so it is not within the scope of the online mall performance analysis and optimization. This paper is designed to optimize performance issues and solutions for online shopping systems:

3.3.1 User experience layer optimization scheme
Reduce static resource loading: reduce HTTP requests as much as possible in front-end optimization, that is, reduce the introduction of js and other static resources.

Front page static: using the page static processing technology, that is, the page does not update for a long time for static processing, delete the static page when updating data. If this static page exists when the user requests it, they will access it directly. If not, they will regenerate the static page.

Front-end framework: at present, front-end development technology develops rapidly and produces many excellent open-source frameworks and js scripts, such as jquery, bootstrap, and easyui. A more mature bootstrap front-end framework was used to replace the native page of the system in the online mall implementation. Makes the front end code clean and beautiful, while avoiding too much duplication of code.

Back-end paging: use back-end paging instead of front-end paging. Page number and page size are passed back through paging plug-ins and asynchronous requests to get data, preventing load delays caused by large data volumes. And use the jquery method to achieve dynamic refresh data, reduce the number of page HTTP requests.

3.3.2 Server layer optimization scheme
Tomcat server optimization The default tomcat is often used as the production environment in the novice deployment. The default tomcat configuration has relatively low memory and threads, which will weaken the server performance and concurrency in the production environment [6-7].

Optimize the tomcat thread: Under the default configuration of tomcat, the minimum free link thread is only 10, the maximum link is 75, and the maximum number of connections is 100. To improve the concurrency performance of the server service, its configuration needs to be modified. Of course, the number of link threads is not infinite, which is also affected by the host operating system kernel parameters. maximum concurrent configuration for Windows is around 2000 and Linux is around 1000. The Linux operating system is currently used by most Java Web servers and server memory issues should also be considered when configuring the tomcat server. In order to ensure the normal operation of the server, the maximum concurrent connection can be set between 600 and 700, and gradually explore the selection of the optimal configuration.

To optimize tomcatIO: Tomcat is a one-connection, one-thread mode. In high concurrency, thread resources are very important. After the thread is allocated, subsequent requests will be blocked, resulting in no response of the page. Therefore, the mode is changed to synchronous non-blocking mode, namely NIO mode, which is synchronized through the selector, reducing thread consumption while avoiding blocking, ensuring serviceability of the service. Its principle is to store data in the buffer when there is a request, and enter the thread operation through the buffer when the thread is idle, thus reducing the time and performance loss caused by the CPU switching IO connection repeatedly.
3.3.3 Nginx proxy optimization

Reverse proxy service is a one-to-many service, whose principle is to achieve clustering purpose by distributing requests [8]. The average Web server is a request for a thread, and the thread in the server is valuable, and when the number of threads is excessive, thread switching will take up a lot of CPU time. But Nginx is a multiplexing mode, Nginx proxy can better improve server performance. Nginx is already an excellent load balancing tool, which can achieve good load effect only by using the default configuration of Nginx. However, for high concurrent projects, it can still be optimized according to the project requirements [9]. For the optimization of Nginx server is mainly reflected in the configuration, and for the online mall system, the required configuration is as follows:

Configure the number of service processes: Because only one process can be processed by a single-core CPU at a certain time, the number of processes can be reasonably configured according to the number of server cores for nginx process, so as to ensure the optimal execution of nginx. Generally, the number of nginx processes is equal to the number of CPU cores, thus reducing the time loss caused by CPU process switching. In the online mall system, the dual-core CPU server is used, so it is most reasonable to configure the number of service processes as 2.

Configure the number of process connections: The number of process connections is the maximum number of processes connected to the Nginx server. Of course, the more the number of process connections is set in theory, the more concurrency Nginx can support. To allow the service to have a better concurrent performance, set the maximum number of processes high, and then test step by step to find the best configuration within the allowable range of server memory. Because this tuning is performed on the local computer, the number of processes is set to be small.

The system opening the number of files at the same time: Because this system is deployed in the Linux operating system, and all services and configurations in Linux are files, it can be regarded as opening a file when a service is used by Nginx, and thus if the number of connections is large, the maximum open file limit error is reported. Therefore, considering the business requirements of the system, increase the number of open files.

Opening sendfile: Sendfile provides fast TCP transfer and reduces CPU overhead because it temporarily stores data in a buffer, it should be turned off in system mode where large file transfer is required. This system is an online shopping system, which does not involve large file transfer, so opening sendfile can improve the response speed of the system.

3.3.4 Data storage layer optimization scheme

Overselling problem: The application system involved in this paper is an online shopping system. According to its nature, the system should ensure the correctness of the inventory when users add goods to the shopping cart and generate orders, to prevent overselling. Also, the online shopping mall needs to deal with high concurrency. According to the previous experience, new problems often arise when the inventory is constrained by adding locks, which leads to a reduction in system performance. Therefore, another optimization scheme should be proposed to deal with this problem.

In terms of increase and decrease inventory and snapping up goods, it is assumed that there are 50 items to be sold, then buffers must be used to avoid frequent read-write databases. In redis, the counter is used to determine whether the user has participated in panic buying. If there is a counter plus one, the user's buying information will be written into the cache. When the counter is 50, all subsequent buying requests will be rejected, to achieve the purpose of peak traffic clipping.

Considering the relatively small concurrent quantity of goods for normal sales, the query and update operation can be adopted to avoid overselling as follows:

‘update goods set count=count-num’Where goodsId=15535 and count-num>=0’

However, in the case of high concurrency, it is dangerous to use this kind of reading and writing, so the principle of solving the oversold problem can also be adopted. The number of goods can be stored in the cache through the set command, and the decrby command can be used for the inventory reduction operation. If the return value is less than zero, the goods is insufficient to carry out the purchase operation. At the same time, the incrby command is used to increase the inventory. Because
redis is atomic, a distributed lock should be added to increase or decrease the inventory to prevent "dirty reads". In this way, the number of database reads and writes can be reduced.

Order frequent operation problem: About the order operation, if the user-generated order and locked the product, but after some time the user stopped buying the product, then the user should release the inventory, should not be allowed to lock the goods for a long time. If the user exists malicious locking of goods, the system and businesses will have a greater impact. In addition, if a large number of users maliciously add or delete orders at the same time, then the system will conduct frequent increase or decrease of inventory, which will greatly reduce the performance of the database. These are all problems that should be avoided in the system. To ensure the operation of the system platform, a reasonable optimization plan should be given for this problem.

For the operation of increasing or decreasing inventory, users' operation must be considered to be irregular. For example, users add goods and generate orders to lock goods, but they do not pay for a long time, resulting in the problem of occupying inventory. Therefore, when the order is generated by the user in the online mall, the user does not directly operate the database. Instead, the operation and order information of the user are stored in the redis cache instead of being permanently stored. After the user completes the payment, the order information is inserted into the database and then the order information is cleared in redis. This not only improves the speed at which users can submit and view orders, but also reduces the pressure on the database from frequent reads and writes.

3.3.5 Code layer optimization

Java provides a lot of high concurrency development interfaces and classes, so in development and optimization it's not just about the performance issues of business logic, but also about the use of the relevant interfaces and methods [10]. On this basis, reasonable use of caching technology in code fragments can also bring good benefits to system performance. The optimization schemes are as follows:

Use the singleton pattern: Singletons are commonly used in software development. Using singletons can reduce the load burden, reduce the time required for loading, and improve the efficiency of the program. However, the singleton model should be used as appropriate. In the process of online mall optimization, the spring framework with low coupling and high extension is adopted, in which the singleton creation instance is realized through dependency injection technology. In addition, singletons can be used in resource control and data sharing to improve system resource utilization and performance. The spring framework is used in the online shopping mall, in which the singleton loading of classes is realized through annotations.

Avoid redundant static variables: In Java applications, static variables are generally not recycled by the garbage collection mechanism. If more static variables are used, these variables will reside in the JVM memory, leading to the reduction of available memory of the system, thus reducing the concurrency of the server. Therefore, the rational use of static variables has a good impact on the efficiency of system operation, so the multiplexing of static variables is achieved through enumeration in the on-line mall.

Avoid building too many Java objects: java language is an object-oriented language, but to reduce system couple and increase system scalability, In java-based system development, new objects should be avoided as much as possible. Therefore, the Java language provides a variety of ways to avoid frequent creation of Java objects, such as factory mode, reflection, dependency injection, and so on. In this Web application system, dependency injection is used instead of the traditional new object, so that the expansion of online shopping mall is improved.

4. Web application system performance optimization test and discussion

After the above system optimization, system performance has been improved in theory. In order to confirm that the above work has indeed improved the performance of the system compared with the traditional system, the system performance test is conducted now [11-12]. The open-source stress testing tool is used in the test, which allows multiple users concurrent access to the Web site and
records the response time for each user's request [13]. And the software can achieve repeated access. Therefore it can be used to simulate user request load, to achieve stress testing, the test before and after optimization is tested in the same machine, and the network environment is the same, so the test has high credibility [14].

4.1 Performance test environment
The system test tool and the web system are deployed on the same machine. The machine configuration is as follows:
- Memory: 2G
- Cpu: intelcorei3 2.4GHz
- The operating system: centos 6.9
- The web server: tomcat 8.05

4.2 Performance test method
First install the siege tool on the Linux test machine, launch the Web site, write the test request path document, and execute the test command. In this test, the document name of the request path is named siege. Url, and then the system's home page, commodity list, commodity details, log in, and registration are respectively stress-tested. Since the system runs on the local computer and the local computer resources are limited, the concurrent amount is set to 100, and the number of repeated requests is set to 10 in this test. In this way, 1000 requests are simulated, and many times of tests are conducted, including home page request, classification request, more commodity request, login request, and registration request. See figure 2.

Figure 2. load test request path.

4.3 Performance test result comparison and analysis
Through the above test example, results were obtained as shown in table 4-1, and the data and results of the main tests in the siege performance test tools were analyzed as follows table 1.

| project                      | Before optimization | after optimization | note                                           |
|------------------------------|---------------------|--------------------|------------------------------------------------|
| total number of completed Transactions(hits) | 20659               | 22857              | total number of transactions processed by the system during the test case |
| success rate of completing the transaction（%） | 95.0~100.0          | 97.0~100.0        | the success rate of the system handling test case transactions |
| Total available (sec) | 170.0–190.0 | 100.0–120.0 | Total time taken to complete the stress test |
|----------------------|-------------|-------------|--------------------------------------------|
| Total size of response data (MB) | 1258.60 | 1263.23 | System test case requests always respond to data size |
| Display network connection speed (sec) | 0.35–0.8 | 0.35–0.8 | Current network latency in system test |
| The average number of transactions completed per second (trans/sec) | 110.0–150.0 | 180.0–190.0 | The average number of request transactions per second processed by the system |

Through the comparison of the above test data, it can be seen that the average performance of the system has been greatly improved after the performance optimization of the system. Due to insufficient computer resources and low test concurrency, there is a small gap between the old and new systems in this test. However, in the case of high concurrency, the optimized system has a strong expansibility.

5. Conclusion
This paper analyzed the factors that affect the performance of traditional web application system. This paper analyzes the factors affecting the performance of the traditional web application system in the case of a large number of users accessing the system concurrently through multiple hierarchies. And according to these factors to optimize the online mall system, we put forward a more typical optimization program. From Web front-end page to database index, query statement and cache optimization to load balance of server, the optimization scheme of Web application system under high concurrency is given and implemented in the system. Finally, several load simulation requests were made through the stress test tool, and the results showed that the optimized system performance was significantly improved. Using the optimization scheme, the performance of the system will be greatly improved.

ACKNOWLEDGMENTS
This work is supported by National Natural Science Foundation of China (No.61672471 and 61502436), Fundamental Research Funds for the He’nan Province University (No.17KYYWF0202), plan for Scientific Innovation Talent of Henan Province (No.184200510010) and He’nan Province University science and technology innovation team (No.18IRTSTHN012).

REFERENCES
[1] Yanan Wang, Huarui Wu and Feng Huang, J. 2000. High-performance concurrent Web application system analysis and research. Computer Engineering and Design Optimization. 15, 8 (Nov. 2000), 2976-2981.
[2] D Kimpe, P, Carns and K Harms et al, C. 2012. Aesop: Expressing concurrency in high-performance system software. In Proceedings of IEEE 7th International Conference on Networking Architecture and Storage (The Hague, The Netherlands, April 01 - 06, 2012).
[3] Y Chen, X Fan and W Yang et al, J. 2005. Stage based parallel programming model for high concurrency stateful network services: internals and design principles, International Journal of High Performance Computing and Networking. 3, 1 (Nov. 2005), 33-44.
[4] A. S. Harji, P. A. Buhr and T. Brecht, JComparing high-performance multi-core web-server architectures,” Proceedings of the 5th Annual International Systems and Storage Conference, pp. 1, 2012.

[5] Kimpe D, Carns P and Harms K, et al, “Aesop: Expressing concurrency in high-performance system software,” Networking, Architecture and Storage (NAS), 2012 IEEE 7th International Conference on, pp. 303-312, 2012.

[6] Xiong Q, He X F, and Tang, J. 2014. Design and Implementation of Authentication Server with High Concurrency. Advanced Materials Research. (Nov.2014), 1791-1794.

[7] Abhishek Chandra Prashan Pradhan Renu Tewari Sambit Sahu Prashant Shenoy, J. 2005. An Observation-based Approach towards Self-managing Web Servers. Computer Communications. (Nov.2005), 1-15.

[8] Michał Daszykowski, Sven Serneels, Krzysztof Kaczmarek, PietVan Espen, Christophe Croux and Beata Walczaka, J. 2007. TOMCAT: A MATLAB toolbox for multivariate calibration techniques. Chemometrics and Intelligent Laboratory Systems. 85, 2, (Nov 2007), 269-277.

[9] Chi X, Liu B, Qi N, et al, J. 2012. Web Load Balance and Cache Optimization Design Based Nginx under High-Concurrency Environment. Third International Conference on Digital Manufacturing & Automation. 1029-1032.

[10] T. J. Richardson and R. Urbanke. J. 2001. The capacity of low-density paritycheck codes under message-passing decoding. IEEE Trans. Inform. Theory. 47, (Nov 2001), 599–618.

[11] Kuipers J, Ueda T and Vermaseren J A M. J. 2015. Code optimization in FORM. Computer Physics Communications. 189, 6 (Nov 2015), 1-19.

[12] Weng L L, Cai W D and Ye Y. J. 2012. Research on pressor strategy of Web application system load testing. Journal of Computer Applications. 32, 10 (Nov. 2012), 2973-2976.

[13] Hui B W, Chen M R and Yang D P. J. 2011. Research and application of performance test on Web application system. Journal of Computer Applications. 31, 7 (Nov 2011), 1769-1772.

[14] R. Hu, H. K. Miao and H. W. Liu. J. 2007. An Approach to Modeling Web Applications Integration Testing. (in Chinese). Computer science. 34, 6 (Nov 2007), 253-257.